VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV AUTOMATIZACE A INFORMATIKY
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

NÁVRH A REALIZACE DETEKCE A SLEDOVÁNÍ ZNAČKY PRO ÚČEL ŘÍZENÍ SEMIAUTONOMNÍHO KONVOJE

DESIGN OF MARKER DETECTION AND TRACKING METHOD FOR SEMI-AUTONOMOUS CONVOZ PURPOSE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Patrik Kaura
AUTHOR

VEDOUCÍ PRÁCE — Ing. Michal Růžička
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav automatizace a informatiky
Student:	**Patrik Kaura**
Studijní program:	Strojírenství
Studijní obor:	Aplikovaná informatika a řízení
Vedoucí práce:	**Ing. Michal Růžička**
Akademický rok:	2016/17

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Návrh a realizace detekce a sledování značky pro účel řízení semiautonomního konvoje

Stručná charakteristika problematiky úkolu:

Tato bakalářská práce je zaměřena na návrh a realizaci detekce a sledování specifické značky pro účel řízení semiautonomního konvoje. Semiautonomní konvoj je složen z několik mobilních jednotek, kde právě první je řízena lidským operátorem a ostatní mobilní jednotky kopírují trajektorii první jednotky. Každá z mobilních jednotek je vybavena specifickou značkou, která je umístěna na zadní části mobilní jednotky. Cílem této práce je navrhnout a prakticky realizovat software založený na zpracování obrazu pro detekci a sledování zmíněné značky. Software by měl byt schopen vyhodnotit vzdálenost od značky a úhel mezi osou kamery středem značky. Dalším požadavkem je běh softwaru v reálném čase na platformě raspberry pi 3. Výstup softwaru by měl být připraven pro předání takto získaných dat do ROSu pro další zpracování.

Cíle bakalářské práce:

Seznámit se s knihovnami OpenCV pro programovací jazyk C++.
Prostudovat možnosti detekce objektu, který má charakter 2D značky.
Prostudovat možnosti sledování daného objektu.
Navrhnout značku, která bude snadno detekovatelná v prostředí, ve kterém bude semiautonomní konvoj operovat. Značka by neměla byt zaměnitelná s jiným objektem v prostředí.
Navrhnout a prakticky realizovat detektor značky. Detektor by neměl zaměnit značku s jiným objektem v prostředí. Detektor by měl být invariantní vůči rotaci značky ve všech třech osách.
Navrhnout a prakticky realizovat sledování značky na základě její detekce. Sledování značky by mělo být stejně jako v případě detektoru invariantní vůči rotaci ve všech třech osách.
Na základě získaných dat z detekce a sledování značky určit vzdálenost k ní a úhel odklonu jejího středu od osy kamery.
Navržený software spustit na platformě rapsberry pi 3, tak aby bylo dosaženo minimálně 25 snímků za sekundu. Pokud bude počet snímků nižší, než bylo specifikováno, tak provést optimalizaci softwaru za tímto účelem.
Vytvořit rozhraní pro přenos naměřené vzdálenosti a úhlu odklonu do platformy ROS.
Provést řadu praktických experimentů k ověření správné funkce softwaru. Zejména se to týká počtu snímků za sekundu, spolehlivosti detekce a sledování značky, dále určení vzdálenosti a úhlu odklonu.

Seznam doporučené literatury:

SHERVIN, E. Mastering OpenCV with Practical Computer Vision Projects, Packt Publishing Limited.
KAEHLER, A. Learning OpenCV 3, O'Reilly Media, Inc, USA, 2015, ISBN 9781491937990

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2016/17

V Brně, dne

L. S.

doc. Ing. Radomil Matoušek, Ph.D. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

ABSTRAKT

Tato bakalářská práce je zaměřena na popis a realizaci detekčního a sledovacího systému pasivního prvku pro účely řízení semiautonomního konvoje. První polovina práce se věnuje shrnutí některých řešení konvojů, použitých měřicích zařízení a výčtu některých dostupných knihoven pro zpracování obrazu. Druhá polovina práce se dále věnuje samotnému návrhu, realizaci měření a výsledkům zkušebních měření.

ABSTRACT

This Bachelor's thesis is focused on the description and implementation of the detection and tracking system of the passive marker for the purpose of controlling semi-autonomous platoon. The first part of the thesis summarizes certain convoy solutions, the measuring equipment used by the trucks in the convoy and it lists some of the available image processing libraries. The second part of the thesis deals with the design itself, the measurement and its results.

KLÍČOVÁ SLOVA

Detekce, sledování, měření, zpracování obrazu, OpenCV, Raspberry Pi.

KEYWORDS

Detection, tracking, measurement, image processing, OpenCV, Raspberry Pi.

BIBLIOGRAFICKÁ CITACE

KAURA, P. Návrh a realizace detekce a sledování značky pro účel řízení semiautonomního konvoje, Brno, Vysoké učení technické v Brně, Fakulta strojního inženýrství. 2017, 65 s., Vedoucí bakalářské práce Ing. Michal Růžička.

PODĚKOVÁNÍ

Rád bych touto cestou poděkoval vedoucímu této práce Ing. Michalu Růžičkovi za odbornou pomoc při vedení této práce a poskytnutí cenných rad v oblasti zpracování obrazu a programování. Dále děkuji své rodině za trpělivost a podporu mého studia.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením Ing. Michala Růžičky a s použitím literatury uvedené v seznamu.

V Brně dne 15. 5. 2017 ……

Patrik Kaura

OBSAH

1 ÚVOD

Růst světové populace je jedním z hlavních problémů dnešní společnosti. S tímto rostoucím trendem se pojí i vyšší hustota automobilové dopravy. Proto se mnohá města a větší silniční celky musejí potýkat s otázkou optimalizace provozu na těchto dopravních spojích. Dlouhé kolony, tvořící se na pozemních komunikacích, vyvolávají v lidech frustraci a snižují ziskovost silniční dopravy. Rovněž je zde přesah do ekologické roviny, kdy neustále popojíždějící vozidla v kolonách jsou zdrojem značné dávky emisí skleníkových plynů.

Částečným řešením a zároveň zefektivněním automobilní dopravy může být takzvané seskupení neboli semiautonomní konvoj. Seskupování vozidel do konvojů je jednou z dostupných metod optimalizování provozu na pozemních komunikacích. Princip seskupování vozidel spočívá ve vytvoření předem určené formace za vedoucím vozidlem. Vedoucí vozidlo určuje rychlost konvoje a směr jízdy. Toto vozidlo je pak v teoretické rovině schopno vést neomezený počet následujících vozidel v tzv. závěsu. Následující vozidla tedy pouze sledují a na základě algoritmů odhadují budoucí polohu vedoucích vozidel. Obsluha těchto sledujících vozidel se „teoreticky“ nemusí věnovat řízení vozidla.

Tato skutečnost však naráží na evropskou legislativu, která není uzpůsobena k užívání takto automatizovaných vozidel na pozemních komunikacích, a proto tyto automobily není možno v režimu semiautonomních konvojů dnes provozovat. Vývoji se však věnuje čím dál větší pozornost a s tím se plánuje i změna této poněkud zastaralé legislativy. První testy některých řešení jsou již v běhu a od hotových produktů nás dělí pouze několik málo let. Některá vybraná řešení, především ze segmentu nákladních vozidel jsou shrnuta v následující kapitole věnované rešerši vyvíjených prototypů.

Cílem této bakalářské práce je seznámit se s technologií zpracování obrazových dat pro využití v semiautonomním konvoji. Vytvořená metoda detekce a sledování pasivního prvku ve formě značky na vedoucím vozidle musí být schopna měřit vzájemnou polohu těchto dvou vozidel a tyto naměřené informace dále předávat systémům určeným pro výpočet predikce polohy vedoucího člena konvoje.

2 SEMIAUTONOMNÍ KONVOJ

Vozidla v semiautonomním konvoji tvoří tzv. formaci. V ní jednotlivé členy reagují na okolní podměty a dynamicky mění vzdálenosti mezi sebou. Díky tomuto systému mohou vozidla mezi sebou udržovat minimální vzdálenost a tím ušetřit kapacitu pozemní komunikace. S minimální vzdáleností se pojí další benefit ve formě menší spotřeby paliva. Zdroje uvádějí pokles aerodynamického odporu o 20–25 %. S menší spotřebou se vozidla stávají efektivnější a ekologičtější. Mezi další benefity se řadí vyšší bezpečnost provozu. Vzhledem k rychlosti reakcí automatického asistenčního systému zdroje uvádějí pravděpodobný pokles nehod, které jsou způsobeny lidskou chybou, o 90 %. V následujících částech jsou shrnuty dnes vyvíjené asistenční systémy, které by se v budoucnu mohly stát součástí tzv. Smart Road. [1]

2.1 SARTRE

SARTRE je zkratkou pro Safe Road Trains for the Environment. Jedná se o projekt Evropské komise, který si kladl za cíl vytvoření funkční strategie a technologie, která by umožnila seřazování vozidel na veřejných komunikacích. Projekt byl odstartován v roce 2009 a sdružoval některé výrobce automobilů v čele se společností Volvo. Projekt směřoval k vývoji prototypu systému semiautonomního konvoje pro stávající komunikace a interakci konvoje s ostatními auty. Během projektu byly použity automobily značky Volvo, přesněji modely S60, V60 a XC60. Všechna vozidla byla schopna řízení v konvoji, a to pro rychlosti 90 *km/h* při stálé vzdálenosti čtyř metrů mezi vozidly. Základem tohoto prototypu byl princip opakování pohybů vedoucího vozidla. K dosažení této vlastnosti byla vozidla rozšířena o kamerové systémy, radar a laserové senzory. Mezi hlavní přínosy patří rozšíření interiéru o HMI (Human-Machine Interface) panel s dotykovou vrstvou na obr. 1. Na tomto panelu dostává obsluha informace o konvoji, případně požadavky jiných automobilů na připojení do formace. [2]

Obr. 1: HMI panel v interiéru vozu Volvo XC60 [3]

Dalším přínosem byla řídící jednotka zajišťující komunikaci mezi členy formace. Tato konfigurace byla testována blízko švédského města Gothenburg na standartních komunikacích. Test byl úspěšný, a to již v roce 2011. Poslední test před ukončením projektu byl proveden v roce 2012 ve španělské Barceloně, kde testovací subjekty dosáhly rychlosti 90 km/h při rozestupech šesti metrů. V témže roce byl projekt ukončen. Volvo se dále zaměřuje na vylepšování tohoto systému o funkce vyhnutí se překážkám nebo nutnosti náhlého brždění. [4]

2.2 Daimler Highway Pilot

Highway Pilot (HP) je projekt společnosti Mercedes-Benz, který byl úspěšně testován v roce 2016. Test proběhl na dálnici A52 blízko německého Düsseldorfu. V testu byly zahrnuty tři nákladní vozy společnosti Daimler Trucks & Buses. Tento konvoj měl rozestupy mezi vozidly patnáct metrů místo obvyklých padesáti, které jsou doporučeny pro bezpečnou jízdu. Vozidla byla vybavena systémem stereo kamer, monitorujících prostor před vozidlem a řadou kamer, využitých pro systémy včasného upozornění. Tyto kamery mají za úkol detekovat vodorovné značení na vozovce, chodce a pohybující se objekty. Stereo kamera je dále využita pro detekci a rozeznání svislého značení. Kromě kamerového systému jsou na vozidle umístěny radarové senzory, monitorující boční strany přívěsu a samotného vozu. Data ze všech měřicích zařízení jsou v řídící jednotce porovnávána a vyhodnocována. Dle těchto dat nákladní automobil koriguje svoji polohu v jízdním pruhu a upravuje vzdálenost od vedoucího vozidla. Pro navigaci jsou dále využity 3D mapové podklady a poziční světla na střeše kabiny, které napomáhají navigaci. [5]

Systém zařazení do konvoje je podobný jako u řešení SARTE. Vedoucí vozidlo dostává požadavek na zařazení do kolony. Pro přenos dat se využívá přenosu Wi-Fi signálem. Vedoucí vozidlo požadavek akceptuje a zahajuje fázi spárování. V této fázi proběhne kontrola všech systémů řízení. V případě poruchy některého ze systému, případně znečištění kamerových systémů, je požadavek na zařazení zamítnut. V případě správné funkce všech senzorů je odeslán požadavek na zařazení do formace. Akceptováním tohoto požadavku řidičem vedoucího vozidla je utvořen konvoj a řidič vedoucího vozidla přebírá řízení konvoje. [6]

Obr. 2: Reakce na změnu jízdního pruhu v režimu obsluhy vozidla (HP je vypnut). [7]

Pro případ změny dopravního pruhu se systém výstrahou připomene obsluze vozidla sledujícího a po uplynutí výrobcem dané doby se deaktivuje. Obsluha tak plně přebírá řízení po dobu vyřešení této situace, následně se znovu zařadí do konvoje. V současné době proběhl úspěšný test tohoto řešení v soutěži European Truck Platooning Challange na trase Stuttgart, Rotterdam. [8]

2.3 DAF EcoTwin

Jedná se o projekt firem DAF Trucks, TNO, NXP a Ricardo, které vytvořily konsorcium EcoTwin. Toto řešení má podobné prvky jako řešení od firmy Daimler. Vozidla jsou tedy rovněž vybavena kamerami a radarem. Kamery jsou pouze na čelní straně vozidla a sledují dění před vozidlem. Jedinečná na tomto řešení je komunikace souprav přes technologii Wi-Fi 802.11p, pracující na frekvenci 5,9 GHz. Jedná se o řešení speciálně navržené pro automobilový průmysl. Pomocí přenosu přes Wi-Fi je přenášen obraz vedoucího vozidla k ostatním členům kolony. Obsluha tak může s předstihem reagovat na situaci, která je před vedoucím vozidlem. Přes stejnou síť je řešena i komunikace mezi obsluhou jednotlivých souprav. Řidiči si tak mohou sdělovat informace, například o opuštění formace a přechodu na režim manuálního řízení. Stejně jako u řešení předchozího jsou soupravy schopny společně akcelerovat a brzdit. Dále vůz přebírají řízení a obsluha následujícího vozidla se mu nemusí aktivně věnovat. Konvoj byl úspěšně testován na rozestupy mezi soupravami 0,5 sekundy během soutěže European Truck Platooning Challange na obr. 3. [8, 9]

Obr. 3: Vozidla DAF EcoTwin v soutěži European Truck Platooning z roku 2016 [10]

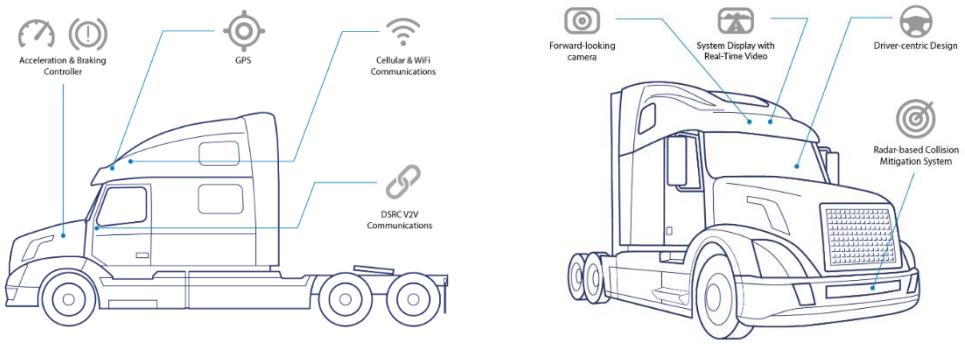
2.4 MAN Truck2Truck

Tento systém byl rovněž předveden v již zmiňované soutěži a je vyvíjen společností MAN Truck & Bus. Stejně jako u systému EcoTwin jsou rozestupy mezi jednotlivými soupravami *0,5* sekundy. Takto malým rozestupem stejně jako předchozí řešení snižují odpor vzduchu a tím i spotřebu paliva. Komunikace mezi vozidly je zprostředkována

pomocí ITS–G5 (Wi-Fi 802.11p). Čelní strana vozidel je vybavena kamerovým systémem, radarem a lidarem. Lidaru se dále práce věnuje v kapitole třetí. [11]

2.5 Peloton Technology

Společnost Peloton Technology Ing. byla založena v roce 2011 skupinou podnikatelů v americkém Silicon Valley. Podobně jako předchozí systémy používá jejich řešení přímou komunikaci mezi vozidly v konvoji. Komunikace ale pouze slouží k synchronizaci akcelerace a brždění souprav. Na rozdíl od předchozích zmiňovaných řešení systém absentuje převzetí kontroly natočení volantu u druhého z vozidel. Každé nákladní vozidlo je vybavené čelní kamerou, radarem, internetovým připojením a komunikací mezi vozidly. Schéma vybavení vozidla je vidět na obr. 4. Celý proces funguje následovně. Společnost Peloton agreguje data o poloze všech svých zákazníků, kteří mají toto řešení nainstalované na svých vozech. Dle pozičních, meteorologických a dopravních informací server nabídne soupravám možnost spárování a zařazení do kolony. Po akceptování požadavku na spárování řidičem vedoucího vozidla dochází k redukci vzdálenosti mezi subjekty. Řídicí jednotka poté pomocí senzorů monitoruje okolní prostor souprav a reaguje na vyskytlé situace. Jak bylo zmíněno, řešení nepřebírá plně řízení, ale pouze část ovládání rychlosti. Řidič druhého vozidla musí stále být aktivní součástí procesu řízení. Z hlediska zabezpečení dat mezi komunikujícími jednotkami je veškerá komunikace šifrována, a tak teoreticky ochráněna před případným zneužitím. V současné době probíhají testy ve Spojených státech. Firma plánuje komerční nasazení v roce 2017, po dokončení potřebných testů. [12, 13]



Obr. 4: Schéma vozidla vybaveného systémem Peloton. [12]

2.6 Autonomous Mobility Appliqué Systém (AMAS)

Systém AMAS je vyvíjen americkou armádou v laboratořích TARDEC (*Tank-Automotive Research, Development and Engineering Center*). Vývoj sady pro autonomní konvoje započal v roce 2012. Cílem je vytvoření sady levných senzorů a systémů řízení na libovolné logistické vozidlo americké armády. Cílem návrhu byla především cena, univerzálnost a systém ovládání tzv. „na jedno tlačítko“. Vozidla v konvoji si mají

udržovat konstantní polohu vůči ostatním zúčastněným a reagovat na nově vzniklé situace na vozovce (vběhnutí civilistů před vozidlo apod.). Řidič si v konvoji může odpočinout a lépe sledovat situaci kolem kolony. Tím se zvýší efektivnost jak vozidel, tak i samotného personálu. První testy proběhly na nákladních automobilech M915, které jsou ve standartní výbavě vojsk. Fotografie vozidla M915 vybaveného systémem AMAS lze vidět na obr. 5.

V testovací sadě je LIDAR, GPS přijímač a snímače pro detekci náklonu nápravy a měření rychlosti kol. Poslední dva zmíněné snímače si laboratoř TARDEC navrhla sama pro tyto specifické účely. První úspěšné testy proběhly v roce 2014 v Jižní Karolině. Vozidla se sadou AMAS byla prověřena řadou zkoušek, při kterých proběhly i nácviky scénářů jednotlivých operací. Tímto testem byl projekt odsouhlasen k dalšímu testování. O rok později Lockheed Martin spolupracující na tomto systému úspěšně otestoval systém CAST (*Convoy Active Safety Technology*). Tento prvek je konstruován s ohledem na zvýšení bezpečnosti. V tomto testu byla vozidla plně zatížena nákladem a operátoři systému museli vyhodnocovat velké množství systému, jako například systém podpory řízení, automatické zastavení před srážkou, adaptivní změnu trajektorie, schopnost udržení vzdálenosti mezi vozidly. Dnes je AMAS vyvíjen firmou Lockheed Martin a je postupně testován i na ostatních vozech. [14]

Obr. 5: Vozidlo M915 vybavené systémem AMAS během testů z roku 2014. [15]

3 SENZORIKA SEMIAUTOMNONÍHO KONVOJE

Semiautonomní konvoje by bez lokalizačních přístrojů nemohly fungovat. Většina dnes vyvíjených vozidel se spoléhá na kamerové a laserové měřicí přístroje. Kamerové systémy většinou pouze snímají okolí vozidel v konvoji. Laserové naopak měří vzdálenosti a všechny tyto informace pak přebírá řídicí jednotka, která dle algoritmu vyhodnocuje danou situaci. V předchozí kapitole byla nastíněna některá dostupná řešení semiautonomního řízení vozidel. Již zmíněné měřicí přístroje v této kapitole budou krátce popsány, včetně jejich principů funkce.

3.1 Senzor

Senzor je zařízení sloužící k převodu fyzikální, chemické nebo biologické veličiny měřené na jinou výstupní veličinu. Stav sledovaného subjektu snímá citlivá část, která sledovanou veličinu promítá do jiného parametru. Výstupní informací je obvykle elektrický signál, který je dále vyhodnocován nejčastěji na elektronické bázi. Senzory se dělí dle výstupní veličiny na senzory s výstupem:

- Elektrickým
- Optickým
- Mechanickým
- Atd.

Dále se senzory dělí dle styku senzoru s měřeným prostředím na:

- Dotykové
- Bezdotykové

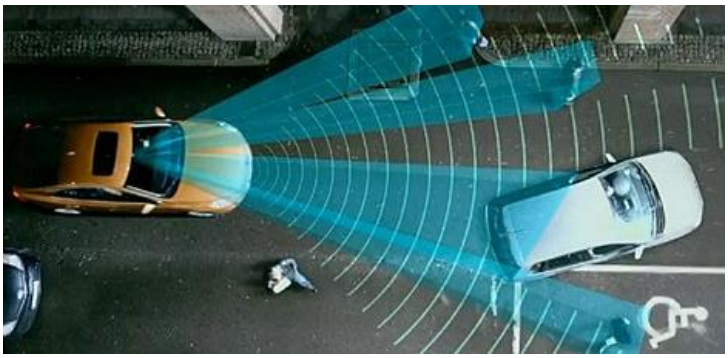
Další vlastnosti a rozdělení senzorů nejsou již účelem této práce, a tak se dále neuvádí. V následujících několika odstavcích jsou krátce shrnuty dnes používané druhy měřicích přístrojů používaných v semiautonomních konvojích.

3.1.1 Laserové dálkoměry

Jedná se o zařízení určené pro měření vzájemné vzdálenosti mezi měřeným objektem a měřicím přístrojem. Většina dnes používaných laserových přístrojů pracuje na principu měření doby letu k měřenému objektu a následnému odrazu zpět do citlivé části přístroje. Nejčastěji obsahují laserové diody nebo LED diody, které jsou zdroji paprsku. Dále je obsažena optická soustava soustřeďující paprsek to tenkého svazku a řada optických filtrů. Přesnost těchto zařízení se pohybuje v rozmezí jednotek milimetrů pro běžné užití. Soustředěný paprsek může být dále zakódován ve formě velmi krátkých pulzů z důvodu zabránění rušení. Existuje i možnost využití Dopplerova efektu, kdy takto řešený dálkoměr dokáže rozeznat pohyb měřeného objektu a následně vyhodnotit rychlost pohybu.

3.1.2 Radar

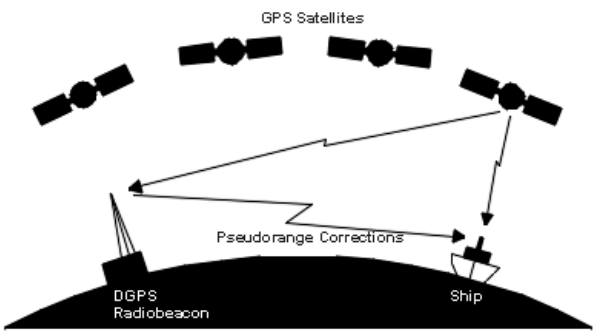
Radar obsahuje vysílač elektromagnetických vln v radiovém nebo mikrovlnném spektru, vysílací anténu a přijímací anténu. Většinou jsou tyto antény sloučeny do jednoho prvku. Dále obsahuje přijímač a jednotku pro zpracování dat. Princip je založen na reflexi elektromagnetických vln. Na rozhraní dvou prostředí s různými dielektrickými konstantami nebo diamagnetickými konstantami se vlna buď odráží, nebo rozptyluje do okolí. Podobně jako u laserových dálkoměrů princip spočívá v měření času mezi vysláním vlny a příjmem odrazu. Tento typ přístrojů byl použit ve všech zmíněných řešeních z předchozí kapitoly. Ukázka použití na vozidle Volvo je na obr. 6.



Obr. 6: Ukázka použití radaru na voze společnosti Volvo. [16]

3.1.3 Diferenciální globální poziční systém DGPS

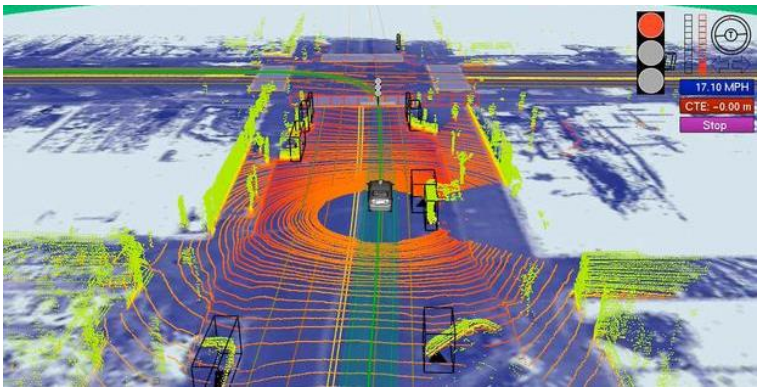
Diferenciální systém lokalizace se používá u většiny zmíněných řešení. Jedná se o vylepšení stávajícího lokalizačního systému s nominální přesností patnácti metrů. DGPS využívá síť fixních referenčních bodů, které vysílají diferenci mezi pozicí předloženou satelity GPS a skutečnou polohou stanice. Schéma lokalizace je uvedena na obr. 7. Vysílání nejčastěji probíhá formou radiových vln v pásmu 285–325 kHz. Takto korigovaná hodnota lokalizace se dostává až na přesnost desítek centimetrů. Z důvodu přesnosti se přijímače používají pouze jako podpůrné jednotky k ostatním měřicím přístrojům. DGPS se kromě semiautonomních konvojů používá v lodní dopravě a v zemědělské technice. [17]



Obr. 7: Ukázka použití stacionární stanice pro lodní dopravu. [17]

3.1.4 LIDAR

Prvek LIDAR (*Light Detection And Ranging*) je velmi podobný již zmíněným laserovým dálkoměrům. Obsahuje zdroj laserového záření, optickou soustavu, mechanický prvek, detektor elektro–magnetického záření a velmi přesné hodiny. Lasery lidaru jsou používány jak v pulsním, tak i kontinuálním režimu s fázovou modulací. Optická soustava lidaru soustřeďuje záření do velmi úzkého svazku. Tento úzký svazek je přiveden na zrcadlo nebo hranol, který je zpravidla umístěn na mechanickém prvku přístroje. Mechanický prvek lidaru s připevněným optickým členem zajišťuje přesné polohování svazku záření. Přesné hodiny jsou důležité z hlediska měření, neboť jsou použity pro měření jednotlivých dob mezi vysláním svazku a odrazem zpět do citlivé části přístroje. U kontinuálních přístrojů je paprsek frekvenčně modulován a vzdálenost může být vyhodnocována i pomocí fázového posunutí přijatého záření. Následující obrázek ukazuje výstupní obraz, který má řídící jednotka automobilu k dispozici od jednotky LIDAR na obr. 8.



Obr. 8: Obrázek ilustruje výstupní data z testování jednotky LIDAR při testech společnosti Google z roku 2011. [18]

3.1.5 Digitální kamery

Digitální videokamery společně s technologií image processing jsou nedílnou součástí semiautonomních konvojů. Dnešní kamery jsou složeny z optické soustavy soustřeďující světelný tok na fotocitlivý prvek. Čidla dnešních kamer jsou řešena buď ve formě CCD (*charge–coupled device*), nebo CMOS, a to v různých variantách. Obě technologie pracují na principu fotocitlivých buněk. Tyto buňky v závislosti na množství dopadajícího světla generují elektrický náboj různé intenzity. O následné zpracování obrazové informace se stará další řídicí elektronika videokamery.

Průmyslově zaměřené kamery jsou konstruovány s přihlédnutím na rychlý záznam obrazového materiálu. Rychlost záznamu měřená v tzv. FPS (*frames per second*) je upřednostněna před rozlišením samotné kamery. Vysoká hodnota je důležitá pro většinu algoritmů zpracování obrazu. Později zmíněný algoritmus optical flow je přímo závislý na vysokém počtu snímaných obrazových vstupů. Mezi další důležité parametry patří velikost snímacího prvku a zorný úhel objektivu kamery.

4 NÁSTROJE ZPRACOVÁNÍ OBRAZOVÝCH DAT

Zpracování obrazových informací je po měření vzdáleností dalším z hlavních prvků semiautonomních konvojů. Snímací přístroj neobsahuje logiku vyhodnocení obrazu, pouze snímky předává řídicí jednotce, kde dle algoritmu jsou vyhodnocována předložená data. Systém řízení z předložených obrazových dat postupně vyhodnocuje informace a následně je předává do samotného algoritmu řízení vozidla. Jednotka tedy z obrazu poskytuje údaje o poloze ostatních objektů vůči vedoucímu vozidlu. Přístroje pro měření vzdálenosti dokáží vyhodnotit polohu v prostoru, ale nedokáží vyhodnotit polohu vůči silničním pruhům nebo vyhodnotit informaci, kterou poskytuje značení v okolí pozemní komunikace. V následující kapitole budou krátce shrnuta některá programová řešení navržená pro zpracování obrazové informace.

4.1 Nástroje pro zpracování obrazových dat

Zpracování obrazu je disciplína vycházející z obecného zpracování signálu. Toto odvětví se soustředí na vývoj počítačových systémů, které jsou schopny pracovat nad předloženými obrazovými daty. Vstupní informace je soustavou algoritmů převedena na výstupní informaci, která je dále zpracována. Nejběžnějším příkladem tohoto odvětví jsou obecně všechny grafické editační programy. Těm se ale práce dále nevěnuje, protože nejsou vhodné pro průmyslovou aplikaci.

4.1.1 OpenCV

Knihovna OpenCV (*Open Source Computer Vision*) je sada funkcí zaměřená hlavně na zpracování obrazu v reálném čase. Sada je multiplatformní a zdarma šířena pod licencí open–source BSD. Programové funkce jsou dostupné v jazycích C++, Python, Java a v rozšíření pro MATLAB/OCTAVE. Dostupná je i hardwarová akcelerace s podporou technologie Nvidia CUDA (*Computer Unified Device Architecture*), kdy se výpočetní výkon přesune z procesoru na grafickou kartu, která je pro tento typ výpočtu výhodnější. Podobnou možnost sdílí i technologie OpenCL (*Open Computing Language*), jež je rovněž součástí knihovny. Samotná sada tedy obsahuje 2500 optimalizovaných algoritmů, které jsou dostupné uživateli. Komunita kolem knihovny je stále aktivní a čítá okolo čtyřiceti sedmi tisíc členů. Hlavní výhodou knihovny je její rozšířenost mezi komunitou a dostupnost dokumentace. [19]

4.1.2 EmguCV

Jedná se o multiplatformní knihovnu v jazyce C#, VB.NET, C++ a IronPython. EmguCV je založena na základě OpenCV již dříve zmíněné sadě. Stejně jako již zmíněná knihovna obsahuje EmguCV sadu matematických metod pro zpracování obrazového materiálu v reálném čase. Rovněž obsahuje rozšíření pro mobilní telefony a technologie paralelních výpočtů za použití grafického jádra. Je dostupná zdarma pro projekty s licencí open

source, v případě zájmu o komerční využití je nutné zaplatit poplatek společnosti EMGU. Hlavní výhodou knihovny je možnost použití vyšších programovacích jazyků. To je zároveň i jistou nevýhodou, protože takto tvořené programy jsou náročnější na použitou technologii. [20]

4.1.3 Intel® IPP

Knihovna z dílen společnosti Intel s názvem IPP (*Integrated Performance Primitives*) je složena ze sady programovacích nástrojů speciálně optimalizovaných pro běh na platformě procesorů Intel. Přesněji je optimalizována pro běh na Intel® Quark™, Intel® Atom™, Intel® Core™, Intel® Xeon™ a Intel® Xeon Phi™. Jak již bylo zmíněno, je speciálně navržena s přihlédnutím na využití specifických instrukčních setů těchto procesorů. Aplikace takto vytvořená je přímo navržena a optimalizována pro danou platformu a je pět až desetkrát rychlejší než aplikace standardně zkompilovaná. Aby byla optimalizace v takové míře možná, je knihovna kompatibilní pouze s jazyky C a C++. Komunitní licence je šířena zdarma pro nekomerční použití. Licence komerční obsahující editor a podporu programovacího jazyka Fortran je již zpoplatněna. Kromě samotného zpracování obrazu knihovna nabízí obecné zpracování signálu, datovou kompresi a kryptografické nástroje. Vybrané části jsou součástí již zmíněné knihovny OpenCV. [21]

4.1.4 SimpleCV

Knihovna SimpleCV je open source projekt sady funkcí pro zpracování obrazového materiálu. Jako u předchozích knihoven vychází její základ ze základů OpenCV. Je navržena speciálně pro jazyk Python a tím je zároveň multiplatformní. Vzhledem ke skutečnosti, že je knihovna napsána hlavně pro jazyk Python, není dostatečně rychlá pro náročné algoritmy. Pro jednoduché zpracování s nižší snímkovací frekvencí je však dostačující. Jazyk Python je tedy zároveň výhodou i nevýhodou této knihovny. Výhoda spočívá ve vysokém komfortu používání, nevýhodou je pomalý běh samotné knihovny při běhu složitých algoritmů. [22]

4.1.5 BoofCV

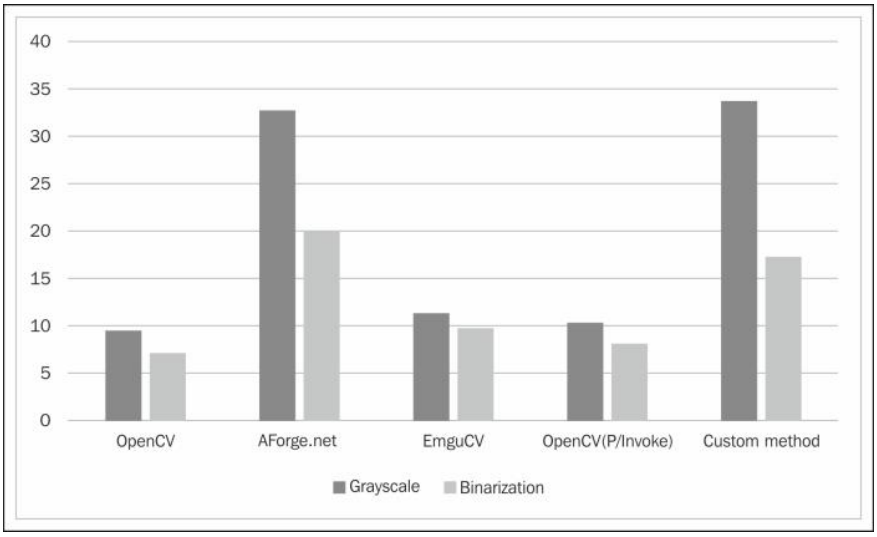
Je knihovna pro programovací jazyk Java a je šířená pod licencí open source (*Apache 2.0*). Je určena ke zpracování obrazu a dalšímu využití v oblasti robotiky. Od začátku je navržena s ohledem na optimalizaci nízkoúrovňových procesů. Skládá se z několika balíků zaměřujících se na zpracování obrazu, geometrickou interpretaci obrazu, kalibraci záznamových zařízení a vizualizaci získaných dat. Dokumentace knihovny je obsáhlá a dostupná na webových stránkách projektu. Výhodou této knihovny je rovněž programátorský komfort z důvodu použití vyššího jazyka, nevýhodou jako u všech knihoven postavených na vyšších jazycích je pomalejší chod v případě složitých algoritmů. [23]

4.1.6 MATLAB Computer Vision System Toolbox

Software MATLAB je nástroj navržený pro řešení inženýrských a vědeckých aplikací. Součástí této aplikační sady je i zmíněný Computer Vision System Toolbox. Ten poskytuje souhrn algoritmů, funkcí a hotových aplikací určených pro simulaci a zpracování obrazových dat v reálném čase. Dále nástroj umožňuje zpracování obrazu ze stereoskopických kamer. To zahrnuje 3D rekonstrukce obrazových dat a 3D bodová projekce dat. Další součástí jsou dnes velice žádané nástroje strojového učení. Programování v této sadě probíhá buď formou strukturovaného textu, nebo formou grafického programování v prostředí Simulink. Mezi výhody tohoto řešení se řadí vysoká univerzálnost sady. Hlavní nevýhodou je jako u již dříve zmíněných knihoven pomalejší chod složitých algoritmu a cena produktu. Cena samotného softwaru MATLAB a jediné nástavby Computer Vision se pohybuje okolo hodnoty 1250 €. [24]

4.2 Volba nástroje

V následující části je shrnuto několik důvodů pro zvolení knihovny OpenCV. Hlavním důvodem použití je možnost psaní kódu v jazyce C/C++. Vzhledem k ostatním dostupným řešením je C/C++ nejvýhodnější volbou pro samotnou optimalizaci chodu. Pro svůj chod nepotřebuje interpretační programy, které zpomalují chod programu jako je to u jazyků Python a prostředí Matlab. Jazyk Java by na platformě Raspberry Pi byl vhodný, ale pro svůj běh potřebuje JVM (Java Virtual Machine), jinak řečeno kompilátor generuje tzv. bajtkód. Ten se pak dále zpracovává a zpomaluje chod. Další výhodou OpenCV je obsáhlá a dostupná dokumentace. Pro zvolení tohoto řešení hovoří i některé výkonnostní testy viz. obr. 9. Z těchto důvodů bylo rozhodnuto zvolení této knihovny.



Obr. 9: Graf z testů výkonosti některých knihoven pro zpracování obrazu. Svislá osa zobrazuje čas v milisekundách potřebný pro provedení operace převodu na černobílý obraz a převodu na binární obraz. [25]

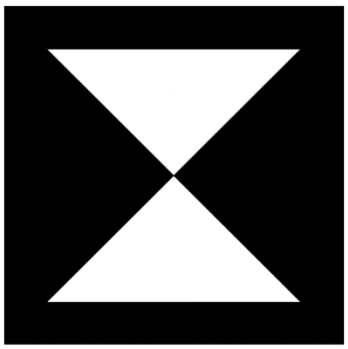
5 NÁVRH ŘEŠENÍ

Cílem této bakalářské práce je navrhnout a prakticky realizovat detekci a sledování specifické značky s cílem poskytnutí dat o poloze vedoucího vozidla kolony. Vstupem je kontinuální přenos obrazových dat pořízených kamerou umístěnou na čelní straně následujícího vozidla. První fáze zvolení softwarové knihovny je popsána v kapitole 4.2, a proto se jí zde práce nevěnuje.

Další fází je návrh samotné značky. Pro práci byla předložena navrhovaná značka vedoucím bakalářské práce. Ta byla v průběhu návrhu pouze drobně poupravena, a to změnou tvaru podkladového materiálu. Původní značka byla umístěna na čtvercovém podkladu. Vzhled původního návrhu může čtenář spatřit na obr. 10. Čtvercový podklad činil těžším detekci značky, a proto byl pouze změněn tento tvar na kruhový. Samotná značka se skládá z černého čtverce a dvou symetricky umístěných trojúhelníků. Návrh detekce této značky se popisuje v následujících několika podkapitolách.

5.1 Návrh algoritmu detekce a sledování značky

V následujícím textu jsou krátce shrnuty jednotlivé kroky navrženého algoritmu. Algoritmus začíná svoji práci nejdříve sejmutím obrazových dat z kamery a následně je převádí do vhodné podoby. Celý program byl sestaven ze dvou fází. Jedná se o fázi detekce značky, kdy algoritmus pomocí předložených funkcí hledá značku v obrazových datech. V druhé části programu, po nalezení dané značky, jsou jednotlivé kontury značky aproximovány a předávány do funkce pro samotné sledování značky v obraze.



Obr. 10: Ukázka původního návrhu značky určené pro detekci vedoucího vozidla.

5.1.1 Detekce hran

Detekce hran je první fází zmíněného algoritmu. Hrana je vymezena mezi dvěma oblastmi s různou hodnotou stupně šedi. Hrana se detekuje tam, kde je rozdíl stupně šedi největší vzhledem k poskytnutým prahům citlivosti. Vzhledem k požadavku černobílého obrazu je nutné nejdříve poskytnutá data převést do této formy. Bez tohoto převodu by detekce hran nebyla možná. Jak již bylo zmíněno, práh citlivosti algoritmu je velice důležitou součástí. Pokud je tato hodnota příliš malá, algoritmus vyhodnotí pouze největší

přechody a obraz neposkytuje prakticky žádné informace. Pokud je tato hodnota naopak příliš velká, výstupní obraz je poté přesycen redundantními hranami. Tato redundance je způsobena detekcí šumu a vad obrazových dat. Na výstupu je binární obraz obsahující filtrované hrany. [26]

5.1.2 Aproximace hran

Tato část má za úkol zjednodušit předložená data pro následné třídění kontur. Data z předchozí části obsahují značnou část neostrých kontur ve formě bílé barvy v binárním obrázku. Pro další kvalifikaci dat je nutno tyto neostré hrany aproximovat proložením úsečkami. Aproximací hran vzniknou geometrické obrazce, u kterých poté můžeme vypočítat plochu, kterou ohraničují. Dále je možno vypočítat další parametry, jako je těžiště, nebo ověřit konvexnost daného obrazce. Výstupem této fáze je binární obrázek jako v předchozím případě, ale tentokrát je zjednodušen na jednoduché tvary.

5.1.3 Filtrování kontur

Filtrování je poslední fází, ve které se z binárního obrazového vstupu vyselektuje hledaná značka. V první části jsou vyselektovány kontury, jež zaujímají v prostoru určitou oblast. Dle velikosti plochy jsou tak odstraněny všechny parazitní kontury, které vzniknou vlivem digitálního šumu. Dále jsou selektovány obrazce dle úhlů, které svírají. Následuje část selekce pomocí počtu hran objektů. Selektují se objekty, jež mají tvar, a tedy i počet hran stejný jako trojúhelník a čtverec. V následující části se vyselektuje pouze tvar, který je čtverec a má uvnitř sebe umístěn alespoň jeden trojúhelník. Předpoklad alespoň jednoho trojúhelníku vychází z možnosti špatného detekování značky. Takto nalezená značka je poté uložena k dalšímu zpracování.

5.1.4 Sledování značky

Sledovacímu algoritmu je v této fázi předložena nalezená kontura. Z této jsou vybrány pouze specifické body. Tyto specifické body se nachází v rozích použité značky. Použitý algoritmus pak na každém snímku vyhledá již nalezené body. V tomto úseku za normálních okolností není předpokládán opětovný chod předchozích částí. V případě chyby detekce nebo celkového zmizení detekované značky z obrazu se algoritmus vrací zpět do fáze detekce značky. Tato část je kritická pro celkový chod semiautonomního konvoje. Je na ní závislá celá část navigace, která ale není součástí této práce. V této části je měřena i výška detekované značky. Tato informace je dále zpracovávána za účelem měření vzdálenosti mezi vozidly. Výška značky je uvažována z důvodu relativní neměnnosti. Při zatáčení vedoucího vozidla dochází ke změně šířkového parametru značky z pohledu kamery vozidla sledujícího. Výškový parametr je měněn pouze se změnou úhlu stoupání vedoucího vozidla. Existence předpokladu jízdy po rovném povrchu však změnu tohoto parametru vylučuje.

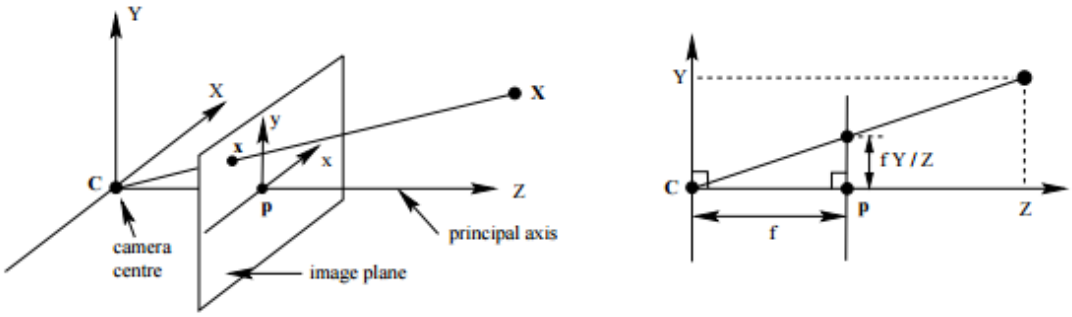
5.1.5 Měření vzdálenosti

Vzhledem k technickým omezením bylo nutné měřit vzdálenost pouze pomocí jediné webové kamery umístěné na čelní straně vozidla. Z předchozí části je změřena aktuální výška detekované značky v pixelech. Tato hodnota je jediná informace získatelná z obrazových dat. Pro určení úhlu mezi osou kamery a značkou je nutné s určitou přesností změřit vzdálenost mezi vozidly.

Princip detekce vychází z principu středové projekce. Model je také v literatuře označován jako model dírkové komory (*pinhole camera model*). Výpočet předpokládá existenci jediného středového bodu, ze kterého vychází celá projekce. Výpočet pracuje v Euklidovském souřadnicovém systému. Model předpokládá centrální bod C, projekční plochu Z vzdálenou od středového bodu C o hodnotu ohniskové vzdálenosti kamery. Dále předpokládá existenci bodu s konečnou hodnotou souřadnic dle vztahu (1). [27]

$$X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{1}$$

Model ztotožnil projekční plochu s plochou snímače kamery. Bod X v prostoru a jeho postavení vůči projekční ploše zobrazuje následující schéma na obr. 11. Z následujících geometrických informací vyplývá jednoduchý matematický vzorec pro určení polohy bodu X vůči projekční ploše. Ve výpočtu je zahrnut parametr ohniskové vzdálenosti.



Obr. 11: Schéma středové projekce použité pro měření vzdálenosti. Obecné schéma (vlevo), zjednodušení pro modelové účely (vpravo). [27]

Tento parametr bude dále rozvinut v samotném řešení, pro konkrétní použitou kameru. Pro samotný výpočet (2) je však nutný již od počátku. Poloha výsledného bodu X se tedy určí. [27]

$$X = (x, y, z)^T \rightarrow (fx/z\,, fy/z, f)^T \tag{2}$$

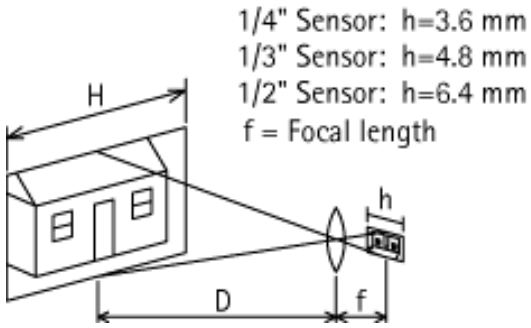
Pro výpočetní účely se rovnice (3) zjednodušuje pouze na jednorozměrný model. Důvodem je pouze jediný neměnný rozměrový parametr detekované značky. [27]

$$\frac{x}{f} = \frac{X}{d} \rightarrow d = \frac{Xf}{x} \tag{3}$$

Kde hodnota x je velikost objektu na projekční ploše. To znamená velikost výšky detekované značky měřené ze získané kontury v pixelech. Hodnota f je, jak již bylo zmíněno, ohnisková vzdálenost kamery. Hodnota X je skutečná velikost detekované značky. Ta je po vytištění v papírové formě na hodnotě výšky 119 cm. Poslední hodnota d je vzdálenost bodu od projekční plochy, tedy objektiv kamery.

5.1.6 Měření úhlu

Pro měření úhlu se vychází ze stejného principu středové projekce. Ze znalosti vzdálenosti, ohniskové vzdálenosti kamery a šířky snímače je algoritmus schopen vypočítat hodnotu šířky H, kterou objektiv kamery zaznamená v obrazových snímcích. Celý princip ilustruje následující schéma na obr. 12. Ze známé hodnoty šířky H a znalosti polohy zobrazené značky je pomocí jednoduché trigonometrie možné spočíst úhel vůči středové ose kamery. Před samotným výpočtem je nutné přepočíst hodnotu H na rozměr jednoho pixelu. [28]



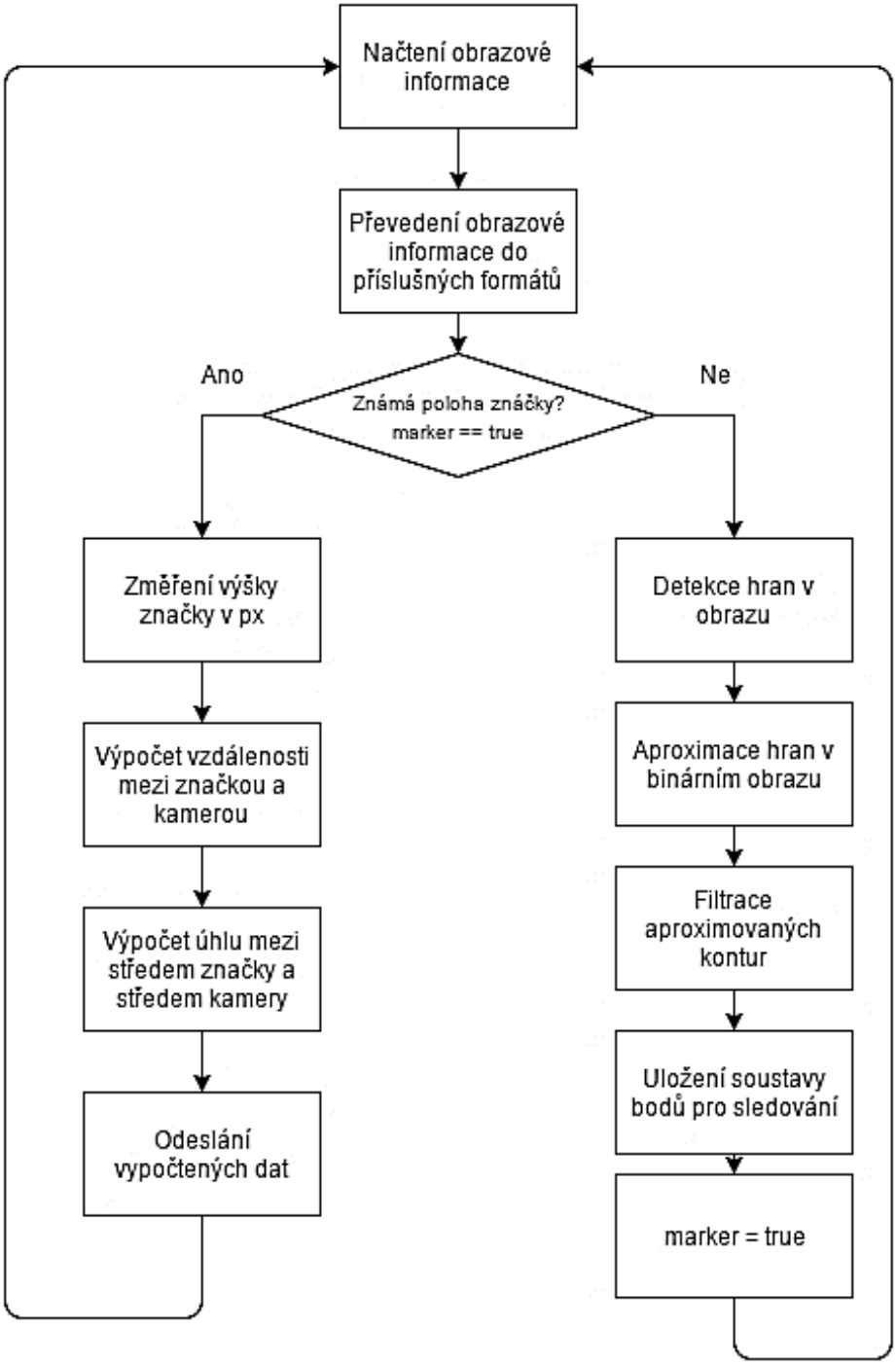
Obr. 12: Schéma středového zobrazení webové kamery a výpočtu šířky H ze známé velikosti snímacího prvku a ohniskové vzdálenosti. [28]

Hodnota P_x v následujícím vzorci (4) značí přepočítanou polohu na detekované značky, hodnota d je změřená vzdálenost vůči projekční ploše (objektiv kamery). Hodnota je již převedena na úhlové stupně pro jednodušší reprezentaci dat. [28]

$$\alpha = \tan^{-1}\left(\frac{P_x}{d}\right)\frac{180°}{\pi} \tag{4}$$

5.2 Zjednodušené blokové schéma

Následující zjednodušené blokové schéma obr.13 zobrazuje algoritmus pro detekci a sledování značky ve fázi návrhu.



Obr. 13: Zjednodušené blokové schéma algoritmu detekce a sledování značky.

6 PRAKTICKÁ REALIZACE

V této kapitole se práce podrobně věnuje samotné realizaci projektu. První část kapitoly bude věnována postupu instalace a samotnému začlenění prostředí knihovny OpenCV do poskytnuté platformy Raspberry Pi. Dále je popisována část vybraných metod detekce a zpracování obrazu. Poslední podkapitola se zaměřuje na distribuci změřených dat a jednoduchou optimalizaci chodu celého programu.

6.1 Raspberry Pi 3

Počítač Raspberry Pi 3 se řadí mezi tzv. SBC (*single-board computer*). Projekt vznikl ve Velké Británii za účelem podpory výuky počítačových věd ve školách rozvojových zemí. Dle asociace Raspberry Pi Foundation bylo prodáno přes deset milionů kusů tohoto zařízení. Pro účely práce byla původně poskytnuta verze Raspberry Pi 2. Tato verze po konzultaci s vedoucím práce byla nahrazena novější verzí třetí (na obr. 14). Důvodem pro opuštění druhé verze počítače je její nedostatečný výkon pro běh samotného algoritmu. S přihlédnutím k dalším projektům, které se mají zpracovávat zároveň s během kamerového systému, byla volba výkonnějšího vybavení nutná. Následující tabulka tab.1 shrnuje některé technické specifikace zařízení. [29]

Tab. 1: Tabulka vybraných vlastností počítače Raspberry Pi 3.

Vlastnost	Popis
Datum vydání	Únor 2016
SoC (*System on chip*)	Broadcom BCM2837 (CPU, GPU, DSP, SDRAM)
Procesor (CPU)	1.2 *GHz,* 64–bitový, čtyřjádrový, ARM Cortex–A53
Video (GPU)	Broadcom VideoCore IV @, 400 *MHz* / 300 *MHz*
Paměť (SDRAM)	1 *GB* (sdílená s GPU)
Integrovaná síť	10/100 *Mbit/s* Ethernet, WiFi 802.11n, Bluethooth 4.1
Interní paměť	MicroSDHC slot
Rozměry	85,60 x 56,5 *mm*

Obr. 14: Osobní počítač Raspberry Pi 3 uvedený na trh v roce 2016. [29]

6.1.1 Instalace operačního systému

Pro zadané účely byla zvolena distribuce Ubuntu Mate 16.04. Tato distribuce byla zvolena z důvodu dostupnosti systému ROS (*Robot Operating System*). Dostupnost tohoto systému sice není pro tuto práci nezbytná, ale na tuto práci navazují další, které tuto platformu přímo vyžadují. Instalace na tuto platformu probíhá prostým zápisem obsahu iso souboru na SD kartu zařízení. Byla vybrána možnost instalace pomocí programu Win32 Disk Imager, kde prostou volbou zdrojového obrazu operačního systému a zápisového média se provedl zápis na toto médium. Po nahrání těchto dat na kartu byl systém spuštěn. Při prvním spuštění je nutné nastavit uživatelské účty a další jednoduché konfigurace. Této části se práce dále nevěnuje, protože se jedná o běžnou součást instalace linuxové distribuce dostupné na stránkách distributora.

6.1.2 Instalace OpenCV 3.1.0

Před samotnou instalací je nezbytné provést instalaci některých nástrojů nutných pro běh knihovny. V první fázi bylo doinstalováno několik potřebných knihoven pomocí následujících několika příkazů viz. tab. 2.

Tab. 2: Příkazy pro instalaci doprovodných programů. [30]

Příkazy v terminálu

$ sudo apt-get install build-essential cmake pkg-config
$ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
$ sudo apt-get install libxvidcore-dev libx264-dev libgtk.2.0-dev

Následující série příkazů v okně terminálu provede stažení knihovny, její rozbalení a konfiguraci viz. tab. 3. Ukázku konfigurace lze vidět na obr. 15.

Tab. 3: Příkazy pro konfiguraci instalace knihovny OpenCV. [30]

Příkazy v terminálu

$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.1.0.zip
$ unzip opencv.zip
cd ~/opencv-3.1.0/
$ mkdir build
$ cd build
$ cmake -D CMAKE_BUILD_TYPE=RELEASE \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.1.0/modules \
-D BUILD_EXAMPLES=ON ..

```
pi@pi-desktop:~$ pkg-config --libs opencv
-L/usr/local/lib -lopencv_shape -lopencv_stitching -lopencv_objdetect -lopencv_s
uperres -lopencv_videostab -lopencv_calib3d -lopencv_features2d -lopencv_highgui
 -lopencv_videoio -lopencv_imgcodecs -lopencv_video -lopencv_photo -lopencv_ml -
lopencv_imgproc -lopencv_flann -lopencv_core
pi@pi-desktop:~$
```

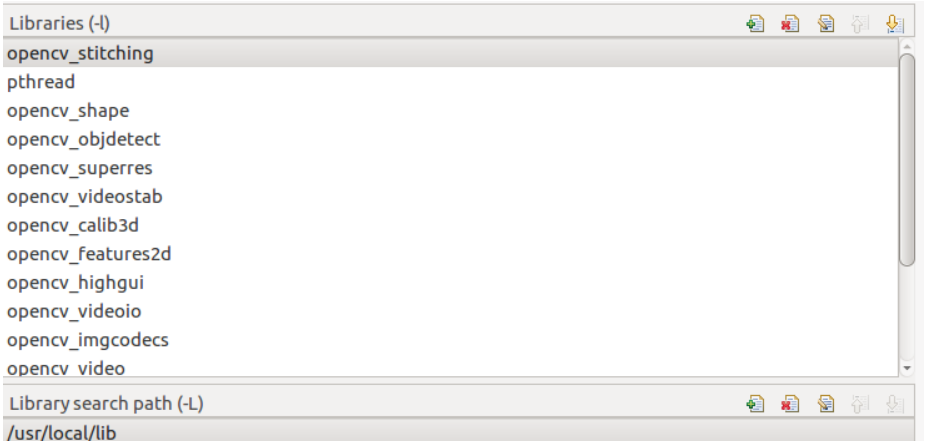
Obr. 15: Konfigurace knihovny OpenCV 3.1.0 pracující na operačním systému Ubuntu MATE.

Fáze konfigurace knihovny je pro chod celého projektu kritická. Při nesprávném nastavení konfiguračního souboru dochází v dané konfiguraci k výpadkům dat z poskytnuté kamery. Dále dochází k haváriím již běžícího programu. Posledním krokem je kompilace knihovny a instalace. [30]

6.2 Konfigurace prostředí Eclipse

Eclipse je multiplatformní IDE (*Integrated Development Environment*) primárně vyvíjeno pro jazyk Java. Díky flexibilnímu návrhu je dnes možné toto prostředí využít i pro jazyky C++, PHP a další. Toto prostředí bylo zvoleno z důvodu vysoké spolehlivosti a velkého množství nastavení. Konkrétně byla použita verze Eclipse Oxygen 4.7, která jako jediná podporuje běh na platformě procesorů ARM. Následující odstavec popisuje konfiguraci projektu pro správný běh knihovny OpenCV. [31]

Po vytvoření projektu pro jazyk C/C++ je nutno daný projekt nakonfigurovat. V záložce Project jsou umístěny Properties daného projektu. V záložce kompilátoru GCC C++ je nutno pro povolení vláken programu nastavit hodnotu -pthread. Dále v záložce Dialect je nastavena verze jazyka C++ na hodnotu ISO C++ 1y, která odpovídá standartu jazyka z roku 2011. Toto nastavení je právě klíčové pro běh vláken programu. V sekci Includes byla nastavena cesta k samotné knihovně. Poslední nastavovanou položkou je začlenění samotných knihoven. Způsob začlenění je možno zhlédnout na snímku z editoru Eclipse obr. 16.



Obr. 16: Nastavení začlenění knihoven do projektu v prostředí Eclipse.

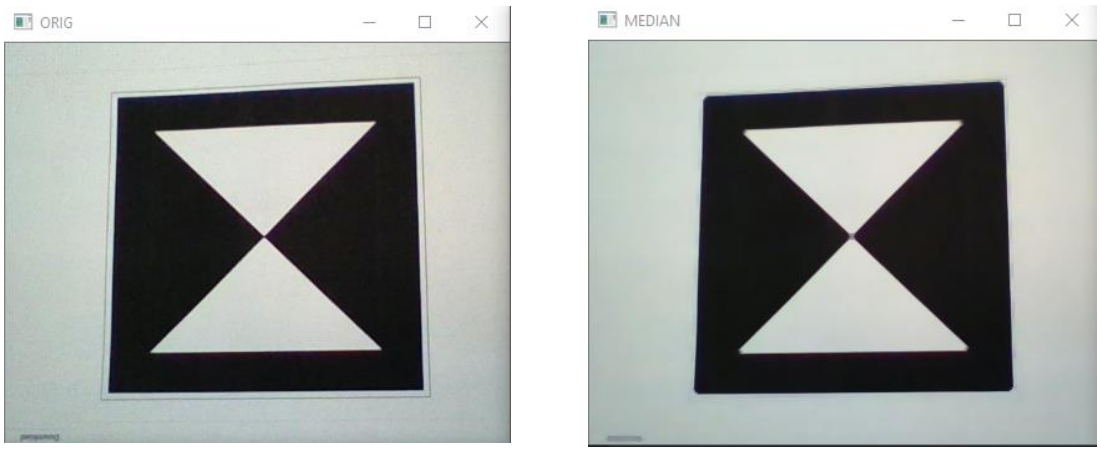
6.3 Použité funkce pro detekci značky

V následujících několika podkapitolách práce shrne použité funkce, z již zmíněné knihovny. U každé z funkcí uvede její princip a ukázku funkce na příkladu obrazových dat. V této podkapitole se práce věnuje pouze metodám pro samotné rozpoznání sledované značky. Metodě následného sledování se věnuje podkapitola následující.

6.3.1 Mediánový filtr

Funkce Mediánového filtru se používá z důvodu částečného odstranění obrazových vad. Jedná se o vady s charakterem šumu, které by samotnou detekce znesnadňovaly. Autor si je vědom výhodnější volby ve formě Bilineárního filtru, tato možnost však na poskytnuté platformě není použitelná z důvodu malého výkonu dané platformy. I při rozlišení obrazu 320x240 *px* není možné používat tento filtr pro aplikace, které se mají vykonávat v reálném čase.

Mediánový filtr není filtrem lineárním, a proto není reprezentován pomocí matice kernelu. Pro výpočet hodnoty daného pixelu operuje filtr s barevnými hodnotami sousedních pixelů. Funkce pak pouze spočítá hodnotu mediánu sousedních pixelů a hodnotu původního nahradí touto mediánovou hodnotou. Díky této jednoduché vlastnosti je filtr efektivní v eliminaci šumů v obrazovém snímku. Dále filtr zachovává hrany a obecně ostrost hran. Nicméně narušuje textury v jednolitých plochách. Funkci mediánového filtru lze zhlédnout na obr. 17. [32]



Obr. 17: Funkce mediánového filtru na ilustrační fotografii. Původní obraz (vlevo), po aplikaci filtru (vpravo).

6.3.2 Adaptivní prahování

Jedná se o modifikovanou techniku prahování. Hodnota prahu je na rozdíl od původní funkce variabilní a mění se v závislosti na zpracovávaných datech. Jedná se o nejjednodušší metodu segmentace obrazu. Používá se pro oddělení oblasti zájmu, která odpovídá sledovanému objektu. Oddělení objektů je možné na základě různé barevné intenzity mezi obrazovými body objektu a pozadí. Takto odděleným plochám je poté přiřazena hodnota 0 (bílá) nebo 255 (černá), a výstupem je tedy binární obrázek. Existuje

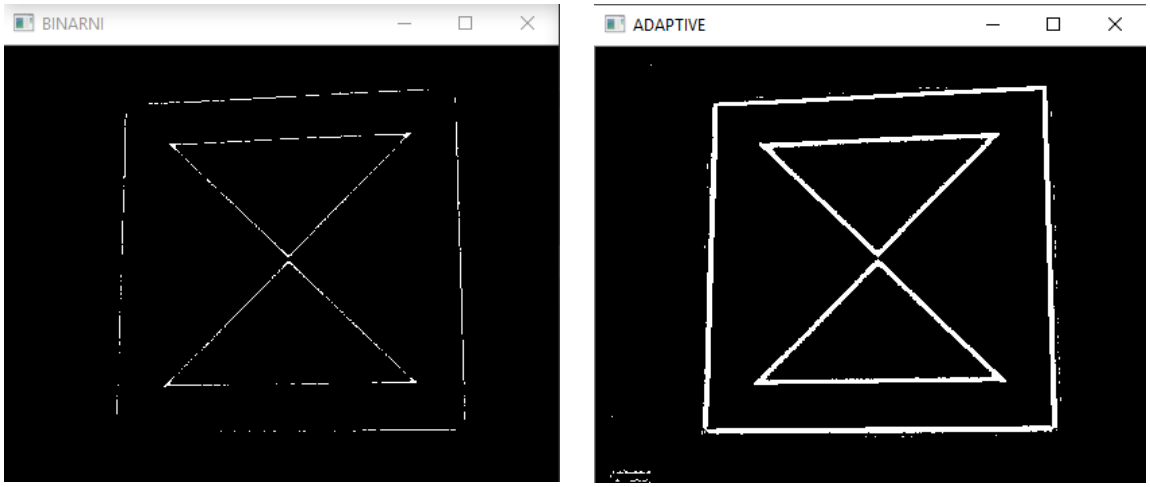
pět základních typů prahovacích operací a funkcí jim přiřazených. Tyto funkce ovšem nebude práce dále popisovat, pouze popíše základní funkci adaptivního prahování. Funkce tedy jak již bylo zmíněno, zpracovává černobílý obrázek a převádí ho na binární. Hodnota je poté počítána pro každý jednotlivý obrazový bod vstupních snímků. Funkce používá dva ze zmíněných pěti typů prahovacích funkcí (tab. 4). [32]

Tab. 4: Funkce Adaptive threshold. [30]

Binární	$\mathbf{dst(x,y)} = \begin{cases} \boldsymbol{maxValue} & \boldsymbol{if\ src(x,y) > T(x,y)} \\ \mathbf{0} & \boldsymbol{else} \end{cases}$
Inverzní binární	$dst(x,y) = \begin{cases} 0 & if\ src(x,y) > T(x,y) \\ maxValue & else \end{cases}$

Hodnota T(x,y) je hodnota prahu vypočítaná pro každý pixel. Hodnota src(x,y) je informace ze vstupního 8–bitového obrazu. Dále se volí funkce, která zajišťuje přizpůsobitelnost. Metoda ADAPTIVE_THRESH_MEAN_C počítá hodnotu T(x,y) z hodnot čtvercového okolí se středem v počítaném obrazovém bodu. Od této hodnoty odečítá konstantu váhy okolí C.

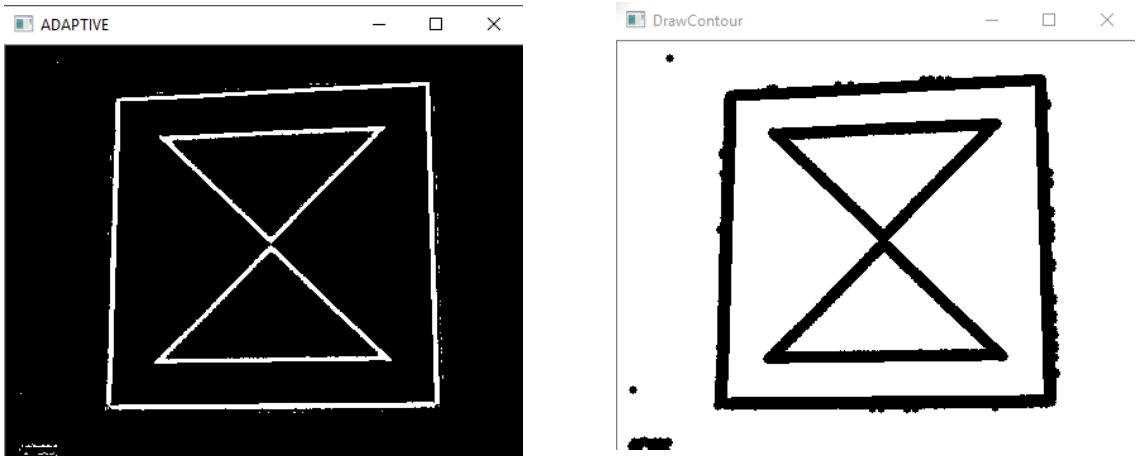
Druhou možností je metoda ADAPTIVE_THRESH_GAUSSIAN_C. Gaussova metoda počítá vážený průměr ze čtvercového okna a odečítá již zmíněnou konstantu C. Funkce pak dále vyžaduje zadání velikosti již zmíněného čtvercového okna. Tato technika je užitečná pro detekci v obrazech se silnou odrazivostí a změnou jasu obrazových dat. Funguje však pouze s 8–bitovými daty nebo s daty s plovoucí desetinnou čárkou. Ukázku a srovnání se standartní metodou binárního prahování lze uvidět na obr. 18. [30]



Obr. 18: Ukázka použití adaptivního (vpravo) a binárního prahování (vlevo).

6.3.3 Hledání kontur

Vstupem do této funkce je již z předchozí funkce vytvořený binární obraz. Kontury jsou v této funkci reprezentovány soustavou bodů. Tato soustava bodu je poté na výstupu reprezentována soustavou vektorů. Hledání kontur vychází z jednoduchého algoritmu, který zahrnuje systematické procházení obrazového snímku. Při dosažení kontury jsou její body zaznamenávány. Po dosažení konce dráhy je znovu spuštěn vyhledávací algoritmus a metoda hledá další kontury. Během průchodu dráhou jsou jednotlivé body zaznamenávány a poté přeneseny do vektorů. Výstupní informace rovněž obsahuje data o hierarchii nalezených kontur, dále jejich zobrazení do grafu. Následující ukázka naznačuje funkci FindCountours v praxi na obr. 19. [32]



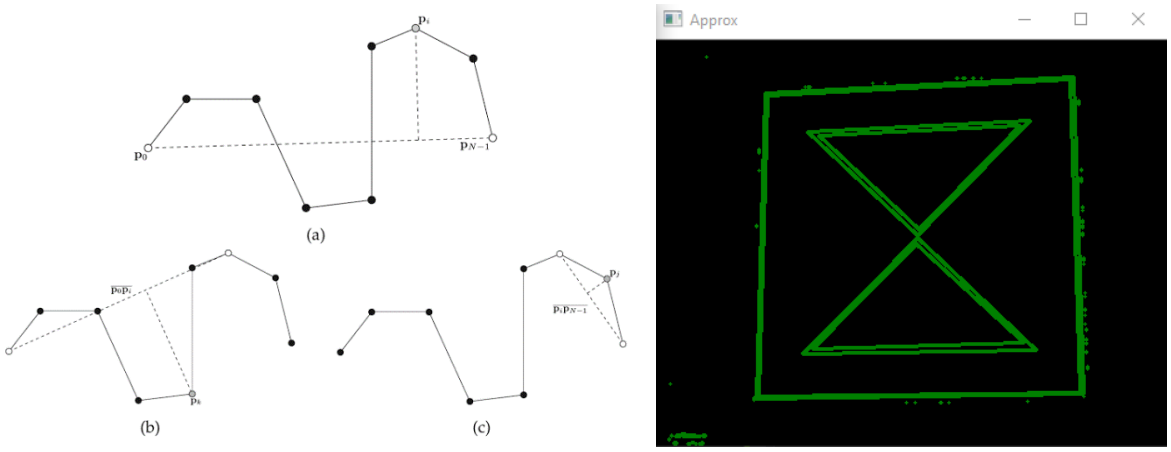
Obr. 19: Ukázka výsledku po aplikaci metody FindCountours. Původní obraz (vlevo) je binární vstup, výsledný výstup složený z vektorů a bodů (vpravo).

6.3.4 Aproximační polynom metodou Douglas–Peucker

Získané kontury jsou již částečně vhodné pro analýzu, ale není možnost počítat jejich hrany a podobně. Výsledný tvar totiž není složen například ze čtyř hran jako standartní čtverec, ale z většího počtu. Tento vyšší počet je způsoben právě nedokonalostí předchozích metod zpracování obrazu. K dosažení lepšího výsledku, který by více odpovídal skutečnosti, je nutné nalezené kontury aproximovat. Metoda uvedená v nadpisu podkapitoly je určena právě pro tyto účely. Výstupem tak bude aproximovaná kontura s určenou přesností aproximace. Aproximace v tomto případě probíhá pomocí algoritmu Ramer–Douglas–Peucker. Jedná se o velmi dobře známou metodu určenou pro redukování počtu bodů křivek. Mějme tedy soustavu bodů a délkový rozměr $\varepsilon > 0$.

V první fázi je zachován první a poslední bod předložené křivky. Následně se postupuje přes všechny body od počátečního bodu do konečného. Metoda započne v počátečním bodě p_0, z tohoto bodu vychází a hledá první bod, který překročí zadanou hodnotu vzdálenosti ε. Od nalezeného bodu p_{0+N} se vrací zpět, směrem k počátečnímu bodu a kontroluje ostatní body, jestli nepřekročí hodnotu vzdálenosti ε. Pokud tuto hodnotu nepřekročí, jsou tedy eliminovány a novým startovním bodem se stává bod p_{0+N}.

Na konci tohoto algoritmu je výsledná aproximovaná kontura. Metoda se vyskytuje předpřipravená v mnoha programovacích knihovnách. Následující série ilustrací ukazuje jednotlivé etapy této metody na obr. 20. [33]



Obr. 20: Ilustrace jednotlivých fází Ramer–Douglas–Peucker algoritmu. Fáze nultá značí samotnou vstupní křivku, fáze čtvrtá značí finální aproximovaný výsledek (vlevo) [33]. Výsledný obrazový výstup aproximované značky (vpravo).

Touto metodou končí podkapitola věnovaná detekci značky na vedoucím vozidle. Způsobu třídění zjednodušených kontur se práce věnovala v kapitole předchozí, přesněji v podkapitole 5.1.3.

6.4 Funkce pro sledování značky

Sledování objektu z obrazových dat se věnuje pouze jedna zde zmíněná metoda. Volbu pouze jedné sledovací metody je nutno použít z důvodu vysoké zátěže zařízení. Při nasazení dalších podpůrných metod program nebyl schopen funkce při požadované hodnotě 25 *Fps*. Avšak i přes použití pouze jedné metody je detekce stabilní a dostatečná pro tyto účely.

6.4.1 Obecný algoritmus Lucas–Kanade

Jedná se o techniku, která slouží pro odhadnutí pohybu objektu zájmu. Mějme vektor změny polohy (u,v), který je přiřazen právě sledovanému obrazovému bodu. Z po sobě jdoucích obrazových rámců lze s touto metodou vypočíst odhad nové polohy sledovaného bodu. Pro správnou funkci jsou nutné tyto předpoklady viz. [34]:

1) Dva po sobě jdoucí obrazové rámce jsou mezi sebou odděleny pouze velmi malým časovým intervalem Δt. Během této velmi malé časové změny se rovněž předpokládá velmi malá změna polohy sledovaného bodu. Z tohoto vyplývá skutečnost, že algoritmus pracuje nejlépe při sledování pomalu se pohybujících objektů.

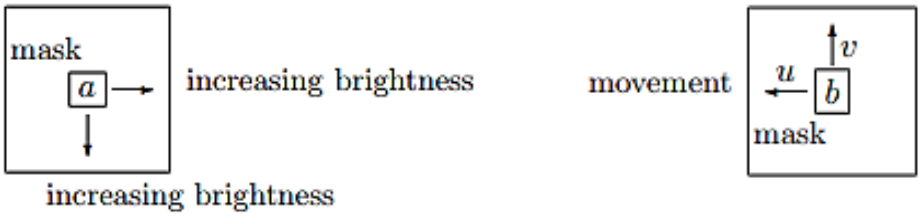
2) Předpoklad určité stálosti jasové intenzity obrazu. Metoda tedy není vhodná na lesklé povrchy s vysokou optickou odrazivostí. Pokud se jasová intenzita nepatrně mění, předpokládá se rovnoměrný průběh její změny.

Metoda pracuje následovně. Předpokládejme scénu, kterou pozorujeme skrze čtvercový otvor s jasovou intenzitou a. Při přechodu na druhý obrazový rámec došlo ke změně právě ve sledovaném otvoru. Intenzita se změnila na hodnotu b. Z tohoto předpokladu můžeme usuzovat změnu polohy sledovaného bodu. Pokud známe změnu intenzity $I_x(x,y)$ pro sledovaný pixel (x,y) ve směru osy x a rovněž známe změnu jasu v ose y tzn. $I_y(x,y)$, pak o celkové změně intenzity jasu při současné změně polohy v ose x o hodnotu u a v ose y o hodnotu v můžeme napsat (5): [34,35]

$$I_x(x,y) \cdot u + I_y(x,y) \cdot v \tag{5}$$

Toto odpovídá lokální diferenci intenzity (b–a), kterou označíme jako $I_t(x,y)$. Tedy můžeme napsat rovnici změny intenzity (6). Tento model ilustruje obrázek obr. 21.

$$I_x(x,y) \cdot u + I_y(x,y) \cdot v = -I_t(x,y) \tag{6}$$



Obr. 21: Schéma ilustruje model čtvercového průhledu obsahujícího sledovaný bod. Původní informace (vlevo) je změnou polohy tzn. změnou intenzity jasu převedena na případ po dokonání pohybu (vpravo). [34]

Protože jediný obrazový bod obvykle neobsahuje dostatek informací o intenzitě, používá se standardně čtvercové pole obsahující okolní body. To znamená, že pro čtvercové pole 3x3 dostáváme 9 lineárních rovnic změny intenzity. Ve značení se zjednodušuje souřadnice bodu x a y jednotným značením, a to bodem p_n pro i = 1...9.

Obecnou soustavu pro n rovnic reprezentuje soustava následujících rovnic (7): [35]

$$\begin{aligned} I_x(p_1)\cdot u + I_y(p_1)\cdot v &= -I_t(p_1) \\ I_x(p_2)\cdot u + I_y(p_2)\cdot v &= -I_t(p_2) \\ &\vdots \\ I_x(p_n)\cdot u + I_y(p_n)\cdot v &= -I_t(p_n) \end{aligned} \tag{7}$$

Soustavu lze maticově zapsat ve tvaru (8):

$$Av = -B \tag{8}$$

$$\begin{bmatrix} I_x(p_1) & \cdots & I_y(p_1) \\ \vdots & \ddots & \vdots \\ I_x(p_n) & \cdots & I_x(p_n) \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} -I_t(p_1) \\ \vdots \\ -I_t(p_n) \end{bmatrix}$$

Tuto soustavu o n rovnicích a dvou neznámých u a v následně algoritmus řeší pomocí metody nejmenších čtverců (9).

$$\sum_{i=1}^{n} \left(I_{xi}u + I_{yi}v + I_{ti}\right)^2 \tag{9}$$

Úpravami dostáváme maticový zápis (10):

$$A^T Av = A^T b \tag{10}$$

Tedy v rozepsané formě (11):

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum {I_{xi}}^2 & \sum I_{xi}I_{yi} \\ \sum I_{xi}I_{yi} & \sum {I_{yi}}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum I_{xi}I_{ti} \\ \sum I_{yi}I_{ti} \end{bmatrix} \tag{11}$$

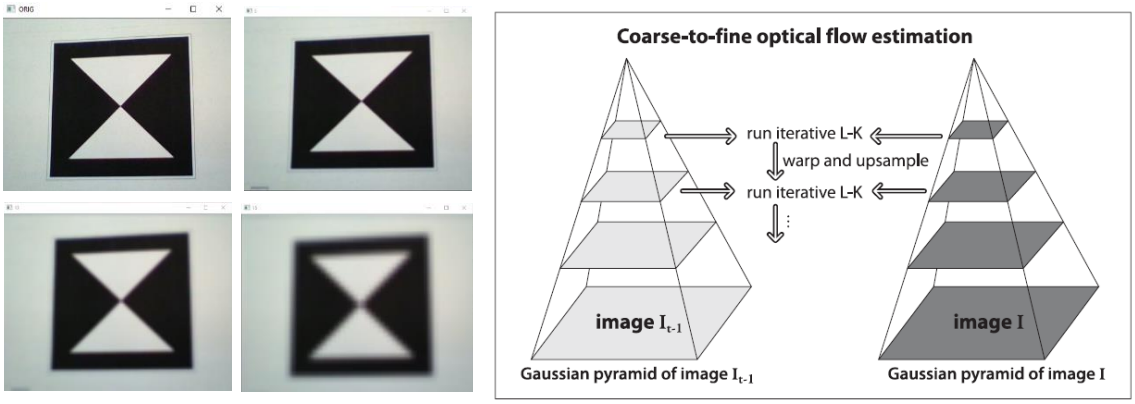
Jednotlivé sumy jsou přes všechny již zmíněné pixely zadaného okna i = 1,2…n. Dále je zahrnuta váhová funkce W(x,y), která jednotlivým bodům přiřazuje určitou váhu. Pak modifikovaná rovnice je upravena do nové formy (12) viz. [34]:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum W^2{I_{xi}}^2 & \sum W^2 I_{xi}I_{yi} \\ \sum W^2 I_{xi}I_{yi} & \sum W^2{I_{yi}}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum W^2 I_{xi}I_{ti} \\ \sum W^2 I_{yi}I_{ti} \end{bmatrix} \tag{12}$$

6.4.2 Optimalizovaný optický tok Lucas–Kanade

Modifikaci základního principu shrnuje následující odstavec. Jak již bylo zmíněno, jedná se o drobnou úpravu, která však přináší výhodu v možnosti detekce rychleji se pohybujících objektů. Původní obrazový rámec je rozvrstven to tzv. pyramidy. Jednotlivé vrstvy této pyramidy tvoří analyzovaný obraz s různou hodnotou rozlišení. Z toho vyplývá, že se tedy jedná o iterační metodu.

V první iteraci algoritmus vypočítá hodnotu vektoru translace pro obraz s nízkou hodnotou rozlišení. Velikost odhadované translace předá do další iterace. Takto předaný vektor slouží jako prvotní odhad posuvu pro druhou iteraci. Druhá iterace znovu probíhá na obraze s vyšším rozlišením. Celý proces se opakuje a hodnota odhadu se stále zpřesňuje. V praxi se používají většinou čtyři vrstvy pyramidy, ojediněle i pět. Z důvodu vícenásobné kalkulace odhadu je tento postup pomalejší než původně zmíněný, avšak je vhodný i pro rychlejší pohyby sledovaných bodů. Následující schémata zobrazují ukázku tohoto postupu na obrazové pyramidě viz. obr. 22. [32,41]



Obr. 22: Vizuální informace o různých hodnotách rozlišení (vlevo) umístěna v jednotlivých patrech Gaussovské pyramidy (vpravo). [32]

6.5 Problematika předávání dat

V poslední fázi musí program předat data o poloze značky další části, která zařizuje integraci těchto dat do systému ROS. Tato část však nebyla součástí zadání této práce. Dána je pouze nutnost odesílání těchto dat tomuto programu. Počátečním požadavkem je běh programu v reálném čase. To jinými slovy znamená zasílání dat o vzdálenosti a úhlu minimálně dvacet pětkrát za sekundu. V kódu pro výpočet těchto parametrů byl použit datový typ float. Tento datový typ v jazyce C++ zabírá paměťové místo o velikosti čtyř bajtů. Bylo rozhodnuto předat data ve formě souboru umístěného v paměti dané platformy. Formát přenosu musel být dostatečně rychle zanalyzován na straně integračního programu. Z důvodů rychlé analýzy dat byl zvolen JSON (JavaScript Object Notation) jako formát pro zasílání dat.

6.5.1 Datový formát JSON

Představuje odlehčený formát pro vzájemné sdílení informace. Je strojově jednoduše analyzovatelný. Založen je na bázi programovacího jazyka JavaScript, přesněji na standartu jeho třetí generace. Struktura je pak složena pouze ze dvou datových struktur. A to z kolekce názvu a hodnoty nazvané též jako objekt, případně struktura. Druhou možností je seznam hodnot. V některých jazycích nazývaný pole, vektor nebo seznam. Tento formát byl zvolen z důvodu jednoduché analýzy vloženého textu. Jazyk Python použitý pro integrační program obsahuje ve svých knihovnách jednoduchý analyzační nástroj dostatečně rychlý pro tyto účely. [36]

Předávání dat funguje následovně. Hodnoty vypočtené ze sledované značky se v paralelně běžícím vlákně převádí do datového formátu JSON. Následně se takto utvořený formát kopíruje do předpřipraveného souboru umístěného na uložišti zařízení. Paralelně spuštěné vlákno se po dokončení odeslání ukončuje a neovlivňuje tak chod hlavního vlákna. Synchronizace čtení a zápisu do souboru se provádí na straně integračního programu. K vytvoření souborů byla využita knihovna JSON for Modern C++. Tvůrcem je Niels Lohmann, který ji uvolnil pod licencí MIT. Knihovna byla vybrána z důvodu dobrých referencí z několika nezávislých testů rychlostí různých knihoven pro tvorbu dat v JSON. Výslednou datovou strukturu, reprezentaci dat a zapisovanou informaci lze shlédnutou v tab. 5.

Tab. 5: Tabulka předávaných dat v několika formátech zápisu.

Příkazy vytvoření	j["A"] = angle; j["D"] = distance;
Zapisovaná data	{"A":-2.19568872451782,"D":241.1866884545135}
Stromová reprezentace	►root {1} ►array {2} A: -2.19568872451782 D: 241.1866884545135

6.6 Optimalizace kódu

Pro účely optimalizace je citováno ze zdroje Clemson university viz. [37]. Některá aplikovaná pravidla z již uvedeného zdroje zde budou krátce popsána. Mimoto byla aplikována i nastavení kompilátoru zajišťující další optimalizaci výsledného programu. Následujících několik bodů shrnuje použitá pravidla optimalizace.

1) Omezení skoků v programu. Každý skok v programu prodlužuje jeho chod. Proto se kód vyhnul používání rekurze. Místo ní bylo použito iteračních postupů. Dále se zamezilo opakovanému volání určitých funkcí v průběhu cyklů. Každá iterace cyklu tak způsobí programový skok, který rovněž prodlužuje čas.

2) Redukování počtu lokálních proměnných. Standardně jsou lokální proměnné umístěny v zásobníku. Pokud je těchto proměnných malý počet, jsou pro funkci dostupné v registrech. Benefitem je přístup do rychlejší paměti a tím i rychlejší chod funkce.

3) Redukce počtu parametrů vstupujících do funkce. Odvíjí se od stejného důvodu jako u předchozího případu.

4) Pokud od funkce nevyžadujeme návratovou hodnotu, je lepší se deklaraci této hodnoty vyhnout. S tvorbou bezejmenného objektu return se zvyšuje výpočetní čas o dobu tvorby kopie do původního místa.

5) Omezení přetypovávání datových typů. Operace změny datového typu vyžaduje operaci kopírování paměťového bloku. Pro datové typy celočíselné a s plovoucí desetinnou čárkou se používají jiné registry, a proto je nutná tvorba kopie. Rovněž zkrácené datové typy jako char, short stále zaberou celou kapacitu registru.

6) V co nejmenší míře omezit funkce vyhledávající v tabulkových hodnotách. Jedná se nejčastěji o trigonometrické operace, které svou hodnotu určují z tabulek předvypočítaných hodnot. V těchto tabulkách pak hledají svoji hodnotu a tím prodlužují běh programu.

7) Inicializace objektů pouze jednou za chod programu. Jedná se o snahu minimalizace inicializačních časů spojených s vytvářenými objekty. Proto se všechny proměnné vytvořily na začátcích funkce a vyhnulo se tak vytváření nových proměnných a instancí objektů například v průběhu cyklů.

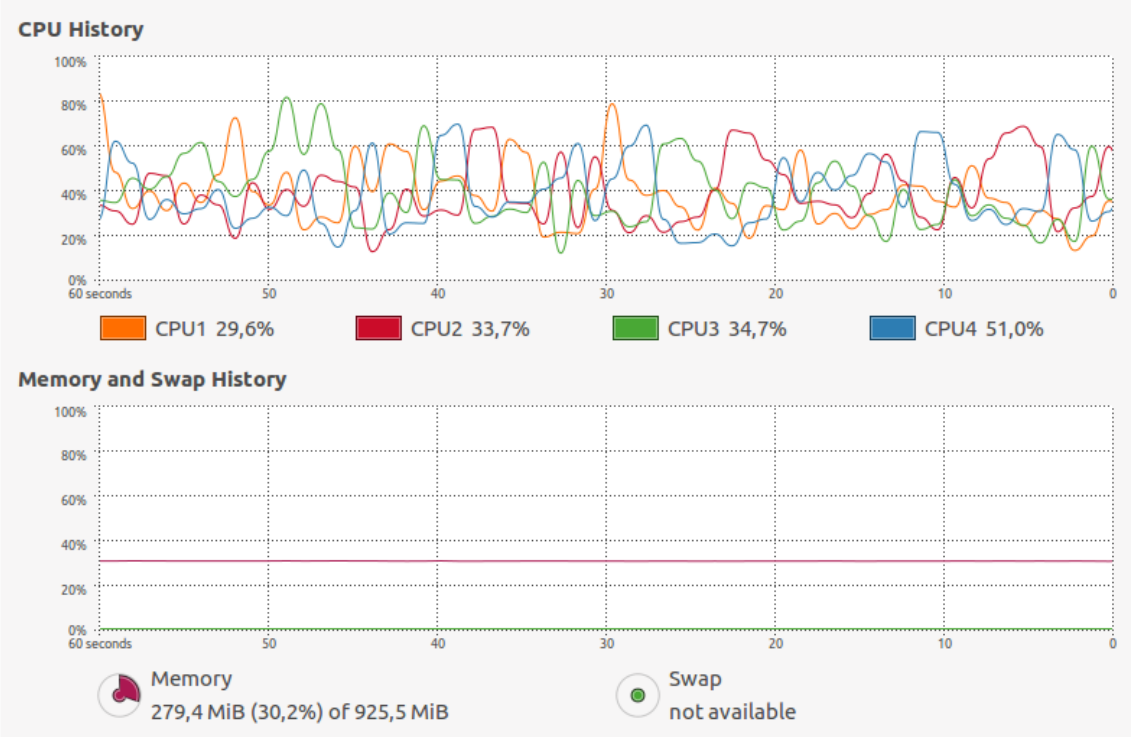
6.7 Optimalizace kompilátoru

Použitý kompilátor GCC ve verzi 4.4 obsahuje několik úrovní optimalizace výsledného programu. Tato optimalizace má za cíl zlepšení výkonu, redukci velikosti nebo zkrácení doby kompilace. Optimalizace na úrovní O0 neobsahuje optimalizaci prakticky žádnou. Kompilátor generuje neoptimalizovaný kód, avšak sestavení programu je poměrně krátké. Na rozdíl od ostatních konkurenčních řešení se jedná opravdu o přesný překlad bez zásahů v oblasti optimalizace. První úroveň úpravy O1 slouží k produkci částečně optimálního kódu, ale bez nutnosti dlouhé kompilace. Dále se sníží doba náběhu programu a velikost dat. Level O2 obsahuje již funkce určené pouze pro některé

procesorové architektury. V tomto módu již nezáleží na velikosti výsledných dat, ale pouze na zvýšení účinnosti programu. Používají se funkce pro rozklad cyklů, převody funkcí na tzv. inline funkce. Výsledný kód je tedy rychlejší a zároveň datově objemnější. Nutno dodat, že druhá úroveň obsahuje všechny funkce z předchozí zmíněné. [38]

Úroveň Os značí speciální druh úprav. Ta obsahuje stejné funkce jako úprava O2, ale nezvětšuje velikost kódu. Na rozdíl od předchozí je kladen důraz právě na velikost výsledného produktu. Konkrétně nepoužívá funkci zarovnání. Zarovnání formátuje funkce, cykly, skoky atd. na paměťové adresy dělitelné dvěma. Jak lze odhadnout, tato změna sice zvýší efektivitu čtení, avšak zvětšuje celkovou použitou paměť. Poslední dostupná možnost optimalizace byla po několika pokusech vybrána pro účely této práce. Příkazem kompilátoru O3 se aktivuje nejvyšší úroveň úprav. V tomto módu se upřednostní rychlost kódu před jeho velikostí. Je zde i aktivována funkce zarovnání i ostatní funkce z O2. Navíc je použita agresivní technika, která zarovnává i všechny podprogramy, se kterými program kooperuje. Nevýhodou této strategie je dlouhá doba kompilace pohybující se okolo jedné minuty a celková velikost souboru. [38]

Velikost je však pouhých 48 *kB*, což vzhledem k paměťovému prostoru karty v zařízení nečiní žádný problém. Samotnými naměřenými parametry se zabývá kapitola následující. Na zde uvedeném snímku z programu System monitor se zobrazuje zatížení zařízení při současném běhu programu na obr. 23.



Obr. 23: Zobrazení zatížení platformy Raspberry Pi. Při testu běhu pracoval program pro zpracování obrazu a operační systém. Graf (nahoře) zobrazuje vytížení procesorových jader, graf (dole) zobrazuje vytížení paměti RAM.

7 PRAKTICKÉ EXPERIMENTY

V této kapitole práce v několika podkapitolách krátce shrne provedená měření na samotném modelu semiautonomního konvoje. První část je zaměřena na popis dostupných technických prostředků poskytnutých pro tvorbu práce. Ve zbylých částech se text soustředí na problematiku měření počtu snímků za sekundu, vývoj teploty CPU a jeho vliv na chod programu a vliv výkonu SD karty na běh programu.

7.1 Kamera Microsoft LifeCam HD–3000

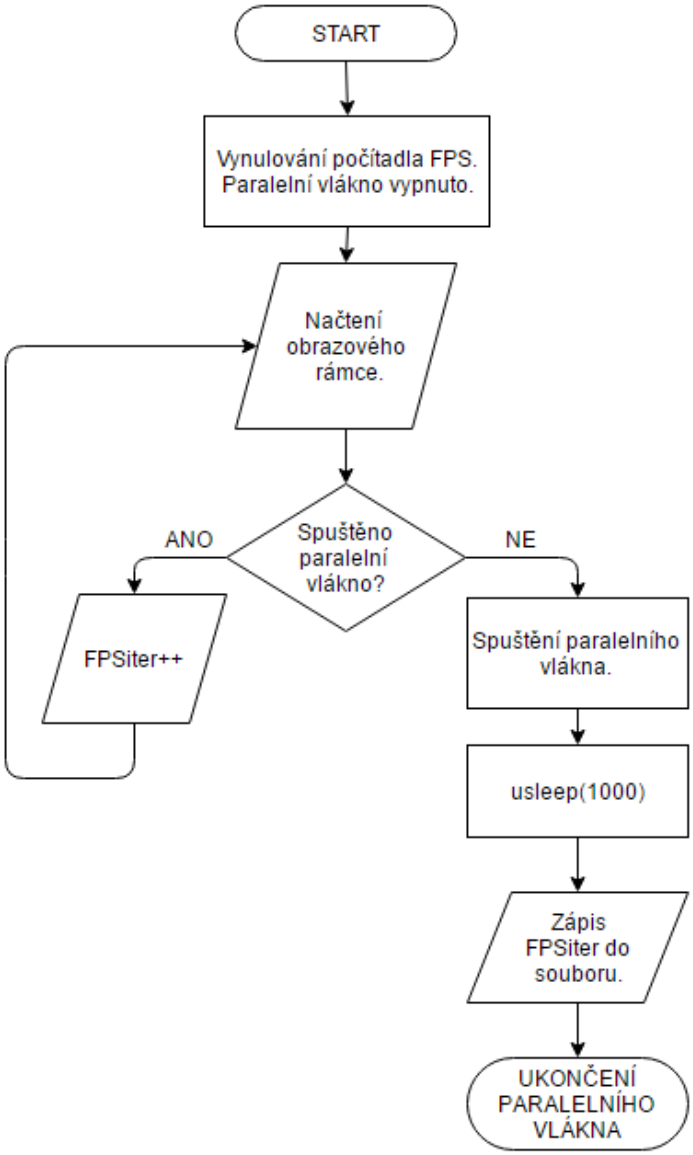
Po konzultaci s vedoucím práce byla vybrána právě tato kamera. V následující tabulce (tab. 6) se uvádí pouze pro tuto práci důležité parametry. Ostatní parametry, jako jsou například nároky na hardwarové vybavení, se zde neuvádí.

Tab. 6: Tabulka vybraných parametrů důležitých pro samotnou práci. [39]

Vlastnost	**Popis**
Technologie senzoru	CMOS
Rozměry senzoru	6 x 4 *mm*
Výstupní rozlišení	1280 x 720 *px* 24 *bit*
Obrazové snímky za sekundu	25 *Fps*
Zorný úhel	68,5°
Pevný ostřící bod	0,3 – 1,5 *m*
Způsob připojení	USB 2.0

7.2 Metodika měření hodnoty FPS

Pro měření hodnoty výstupních snímků za sekundu je v programu použito paralelního vlákna. Celý průběh měření probíhá následovně. Při započtení prvního snímku je do iterační proměnné přičtena jednička. V této iteraci je i spuštěno paralelní vlákno, které je následně uspáno na dobu jedné sekundy. Po této době je paralelně běžící vlákno probuzeno a zapíše aktuální hodnotu proměnné, do níž se během iterací přičítala s každým snímkem hodnota jedničky. Aby s každým obrazovým vstupem nevzniklo nové paralelně běžící vlákno, je v programu zakomponována funkce jednoduchého semaforu, který zamezuje vznik této možnosti. Přesnost doby uspání funkce usleep se u vlákna pohybuje v rozmezí 1–2 ms. Vzhledem k předpokladu běhu programu v maximální rychlosti 40 ms je tato přesnost dostatečná. Následující zjednodušené blokové schéma na obr. 24 zobrazuje tento princip, který byl použit pro měření.

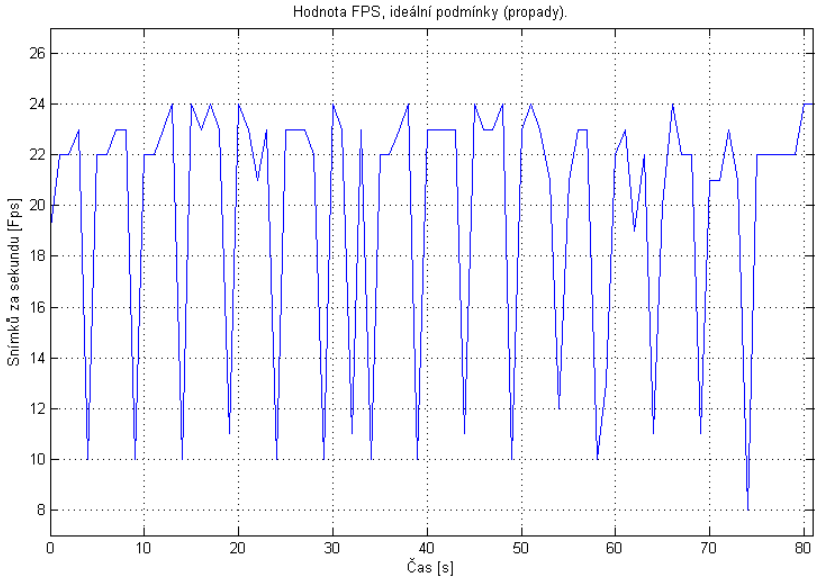


Obr. 24: Zjednodušené schéma programu měřícího hodnotu FPS.

7.3 Prezentace výsledků měření FPS

Tímto způsobem byla provedena dvě měření. První měření se zaměřilo na funkčnost systému detekce při zhoršených světelných podmínkách. Z důvodu technických omezení použité kamery se hodnota počtu snímků pohybuje na úrovni patnácti snímků za sekundu. Tato hodnota je ovšem vynucena samotnou kamerou. Při zhoršených světelných podmínkách potřebuje kamera delší expoziční čas pro dodání jasově vyváženého obrazu. S delším expozičním časem se snižuje i počet snímků za jednu sekundu. I přes tuto skutečnost je obraz jasově vyvážený a pro použití detekce dostačující. V případě druhého měření již světelné podmínky odpovídaly ideálnímu stavu. Toto měření tedy ze strany kamery nebylo nijak omezeno. Jediné omezení vyplývalo ze strany hardwaru, který pro tyto výpočty není primárně určen. Po provedení tohoto měření musela být provedena změna v metodě ukládání dat na SD kartu zařízení. V průběhu měření se v pravidelných

intervalech s rozestupy okolo pěti sekund objevovaly propady hodnoty FPS. Tedy stav, kdy ze stabilní hodnoty 25 snímků docházelo k propadům na hodnotu 10 snímků za sekundu v neměnném prostředí. Tyto propady však nebyly patrné při měření za zhoršených světelných podmínek. Proto se přikročilo ke změně ukládání samotných odesílaných dat. Propady této hodnoty jsou zobrazeny na krátkém testovacím měření na obr. 25.



Obr. 25: Propady naměřené hodnoty FPS při ideálním osvětlení.

7.4 Technologie RAM disk

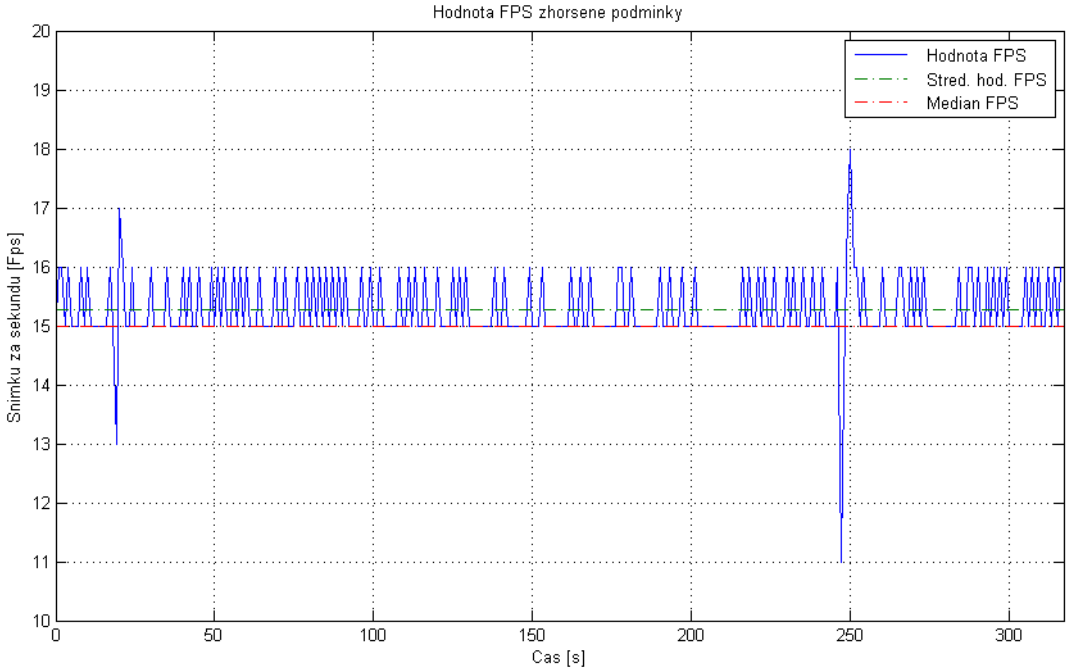
Jedná se o techniku, kdy je v paměti RAM vytvořen virtuální disk, na který má uživatel přístup a může jej obsluhovat. Jedná se o nejjednodušší způsob získání velmi rychlé paměti. Pro účely této práce byl tedy vytvořen kód, který po se po každém startu operačního systému vytvoří oddíl v paměti RAM. Nevýhodou je ovšem zánik všech dat po restartu počítače, kdy se obsah pamětí nenávratně smaže. Tato vlastnost však v tomto případě nečiní žádný problém. Po každém spuštění programu se automaticky maže původní soubor zápisu a vytváří se nový. Následující série příkazů (tab.7) vytváří a formátuje již zmíněný oddíl v operační paměti. Zvolená velikost jednoho megabajtu je pro danou aplikaci dostačující. [40]

Tab. 7: Příkazy terminálo pro vytvoření oddílu v operační paměti. [39]

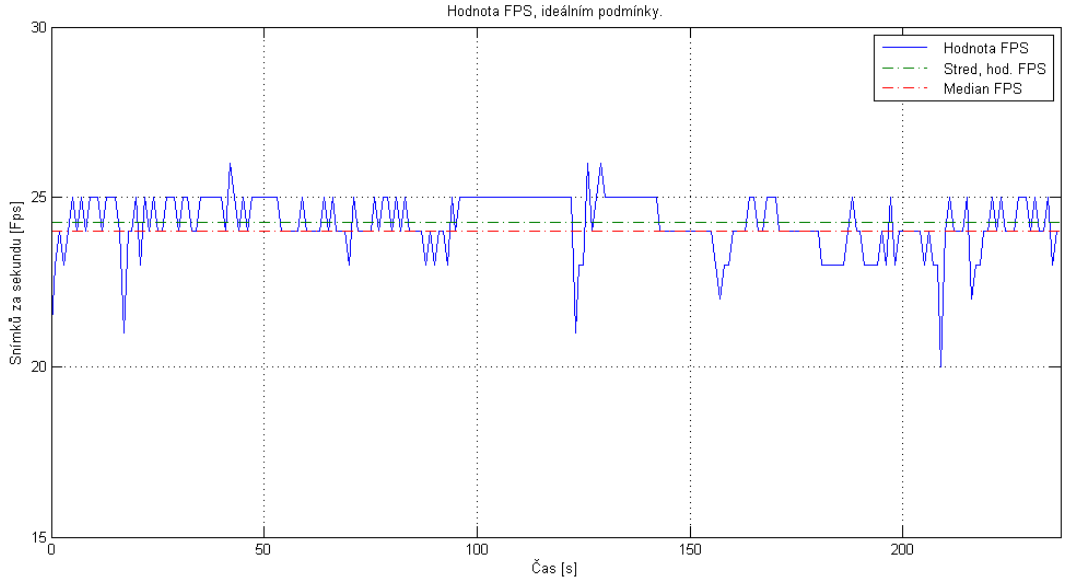
Příkazy v terminálu

$ sudo -s
$ mkdir /tmp/ramdisk;
$ chmod 777 /tmp/ramdisk
$ mount -t tmpfs -o size=1M tmpfs /tmp/ramdisk/

Historii naměřených dat dále může zaznamenávat program na straně integrační. Po vytvoření tohoto disku se provedlo měření při ideálních podmínkách. Následující série dvou grafů (obr. 26,27) zobrazuje naměřená data již s použitou technologií RAM disku. V případě měření ve zhoršených podmínkách se se změnou technologie ukládání výsledky nijak nezměnily. Dostupná SD karta pro tyto rychlosti dostačuje. Ovšem pro běh v maximální rychlosti je nedostačující její zápisová rychlost.



Obr. 26: Graf z měření při zhoršených světelných podmínkách. Střední hodnota FPS je rovna 15.35 a medián je roven 15 snímkům za sekundu.



Obr. 27: Graf z měření při ideálním osvětlení. Střední hodnota FPS je rovna 25 snímkům a hodnota mediánu je rovna 24.35 snímkům za sekundu.

7.5 Vliv teploty na hodnotu FPS

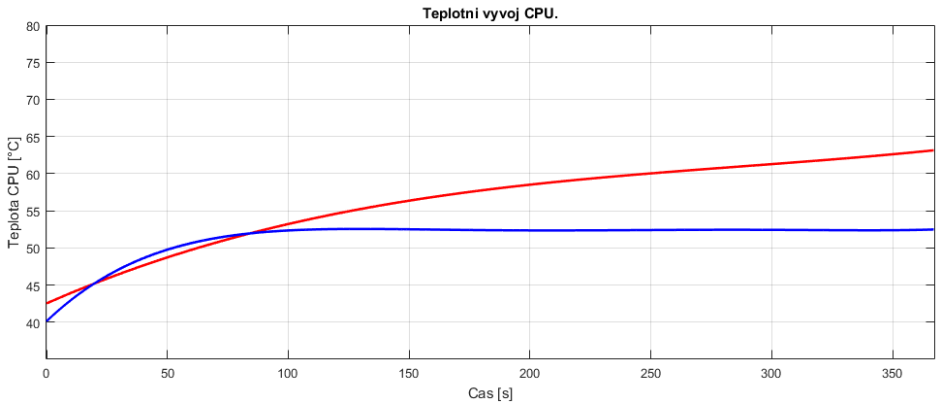
Druhým zásadním faktorem, který ovlivňuje výkon aktuálně běžícího programu, je teplota. U Raspberry Pi 3 je znám problém s přehříváním procesorové jednotky. V průběhu testování bylo nutno zabezpečit dostatečné chlazení procesorové jednotky. Při překročení teplotní hranice šedesáti stupňů jádro automaticky mění svou taktovací frekvenci. Hodnota se tedy pohybuje mezi maximální hodnotou 1,2 GHz a sníženou frekvencí 700 MHz. Správný běh programu je však možný pouze při maximální frekvenci. Proto byl nasazen na pouzdro procesoru hliníkový chladič. Ten sice zvyšuje plochu pro chlazení, ale změna teploty se pohybovala pouze okolo dvou stupňů celsia. Dalším logickým krokem je nasazení aktivního chlazení. Při kombinaci pasivního i aktivního chlazení dochází k udržení teplot na stabilních hodnotách a nedochází ke změně taktovací frekvence jádra.

Pro účely experimentů byl sepsán jednoduchý program (tab.8), který využíval teplotní sondu v jádře procesoru. Mimo záznamů o teplotě program zobrazuje i hodnotu aktuální frekvence jádra. Zaznamenané údaje a ukázku měřícího programu lze zhlédnout pod tímto odstavcem na obr. 28.

Tab. 8: Program pro měření teploty a frekvence procesoru zařízení Raspberry Pi 3.

Program v shellu

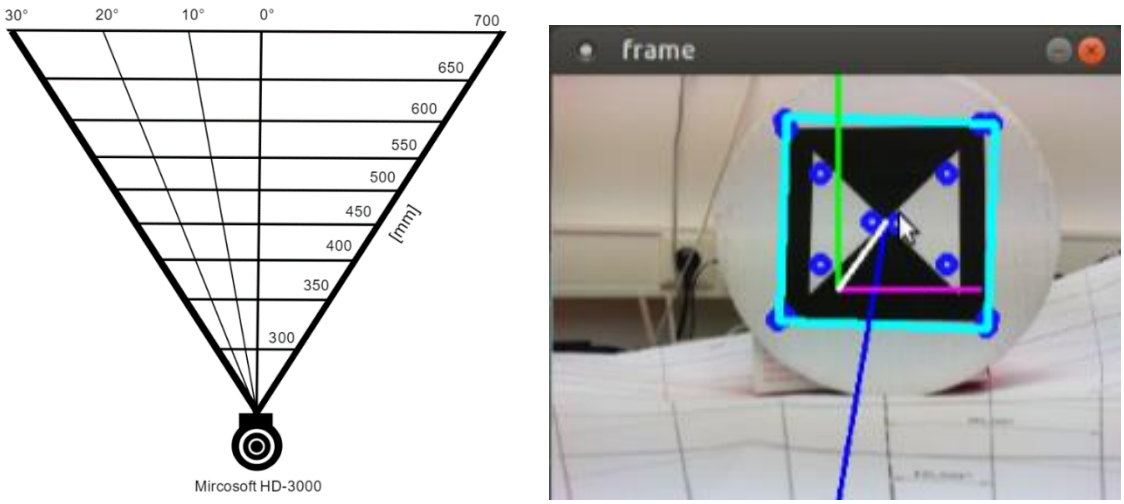
```
#!/bin/bash
while :
do
   temp=`/opt/vc/bin/vcgencmd measure_temp`
   temp=${temp:5:16}
   sleep 1
        sudo cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq
        echo $temp >>/home/pi/tempLog.txt
       done
```



Obr. 28: Teplotní data sesbíraná sondou v jednotce CPU. Modrá křivka značí stav s aktivním a pasivním chlazením současně. Červená čára zobrazuje stav pouze s pasivním chlazením.

7.6 Měření přesnosti detekčního systému

Tato předposlední podkapitola se zabývá určením přesnosti měření vzdálenosti a úhlů. Pro účely měření byla vytištěna předloha s vyznačenými body, které poté byly změřeny a statisticky zpracovány. Během měření byla kamera upevněna na stativu. Tímto uložením byla zajištěna stálost polohy měřicího zařízení. Poté byla značka umístěna na předem zvolené měřicí body a v těchto bodech se provádělo měření. První část testování se zaměřila na měření vzdálenosti. Bylo měřeno v rozsahu 350–700 mm. Mezi jednotlivými body se značka přesunovala o vzdálenost padesáti milimetrů. Měření vzdálenosti se nejdříve provedlo v ose předlohy, tzn. 0°. Poté se na vybraných bodech provedlo opětovné měření vzdálenosti. To se však provedlo mimo osu předlohy. Měřicí rozsah pro určení hodnoty úhlu byl stanoven v rozmezí 0–20° na obě strany. Hodnota 30° nebyla využita, jelikož s tímto úhlem detekční systém zaznamená částečný výpadek značky ze záběru a systém přestane s měřením. Schéma měření s ukázkou praktické realizace zobrazuje obr. 29.



Obr. 29: Schéma měření s osnovou (vlevo). Praktická realizace měření se záběrem z kamery ve vzdálenosti 700 mm (vpravo).

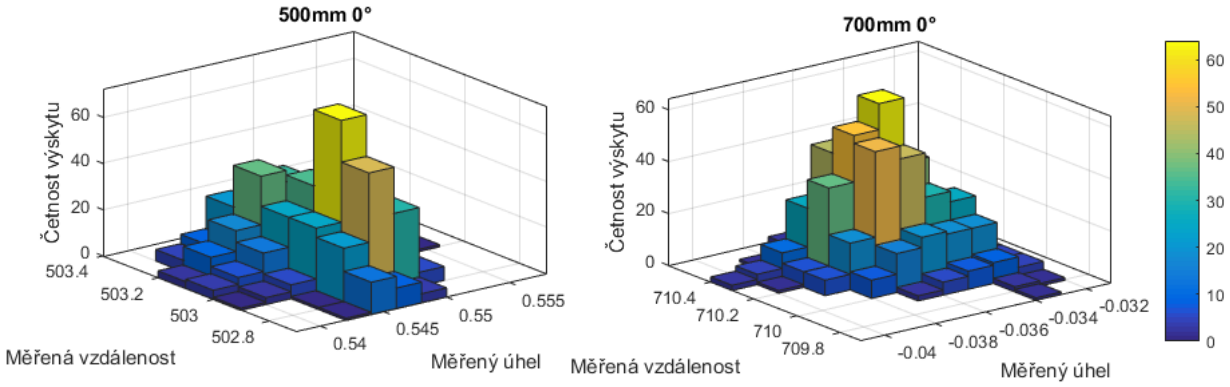
Všechny naměřené parametry byly zaznamenány tisíckrát. Na tomto statistickém souboru byla provedena jednoduchá analýza s určením relativní a absolutní chyby měření. Před samotným měřením bylo nutno nastavit výpočetnímu programu přesné hodnoty ohniskové vzdálenosti čočky kamery a rozměrové parametry CMOS senzoru. Ty byly nastaveny na hodnoty 6x4 mm pro senzor a 3,64 mm pro ohniskovou vzdálenost. Následující série tabulek (tab. 9,10) zobrazuje výsledky jednoduché statistické analýzy. Dále jsou zde zahrnuty vybrané trojrozměrné histogramy, které poukazují na preciznost jednotlivých měření viz. obr. 30,31. Avšak přesnost a preciznost u měření úhlů není tak dobrá se zvětšujícím se úhlem, jednotlivé chyby se přisuzují nedokonalosti měřicí metody. Pro účely semiautonomního konvoje jsou však tato data dostačující.

Tab. 9: Tabulka shrnující výsledky jednotlivých měření vzdáleností. Statistický soubor čítal 3000 vzorků pro každou vzdálenost (měřeno ve třech úhlech).

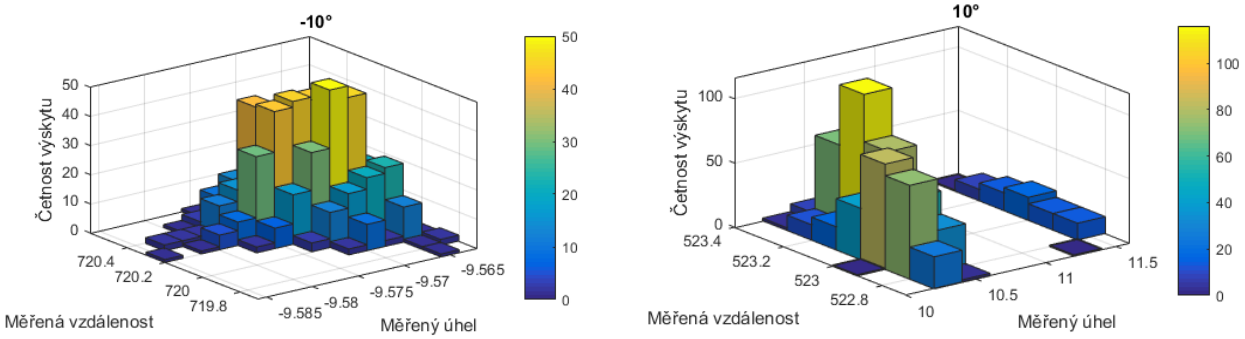
$Dist_{max}$	$Dist_{min}$	Aritmet. průměr	Medián	Rozptyl	Směrodatná odchylka	Absolutní chyba	Relativní chyba
350,896	350,732	350,807	350,805	0,001	0,038	-0,807	-0,231
400,158	399,942	400,044	400,043	0,002	0,050	-0,044	-0,011
455,781	455,781	455,781	455,781	0,000	0,000	-5,781	-1,285
503,354	503,024	503,199	503,201	0,007	0,082	-3,199	-0,640
601,028	600,375	600,717	600,686	0,027	0,163	-0,717	-0,119
651,058	650,403	650,758	650,732	0,019	0,138	-0,758	-0,117
710,259	709,709	710,087	710,100	0,012	0,108	-10,087	-1,441
[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[mm]	[%]

Tab. 10: Tabulka shrnující jednotlivá měření úhlů. Statistický soubor čítal 3000 vzorků pro každou vzdálenost (měřeno ve třech vzdálenostech).

$Úhel_{max}$	$Úhel_{min}$	Aritmet. průměr	Medián	Směrodatná odchylka	Absolutní chyba	Relativní chyba
-0,032	-0,036	-0,034	-0,034	0,001	0,034	
0,556	0,548	0,552	0,552	0,002	-0,552	
-9,562	-9,570	-9,566	-9,566	0,002	-0,434	4,341
11,477	10,358	10,574	10,434	0,367	-0,575	-5,747
-20,747	-20,755	-20,751	-20,751	0,002	0,751	-3,754
21,678	21,676	21,677	21,677	0,001	-1,677	-8,384
[°]	[°]	[°]	[°]	[°]	[°]	[%]



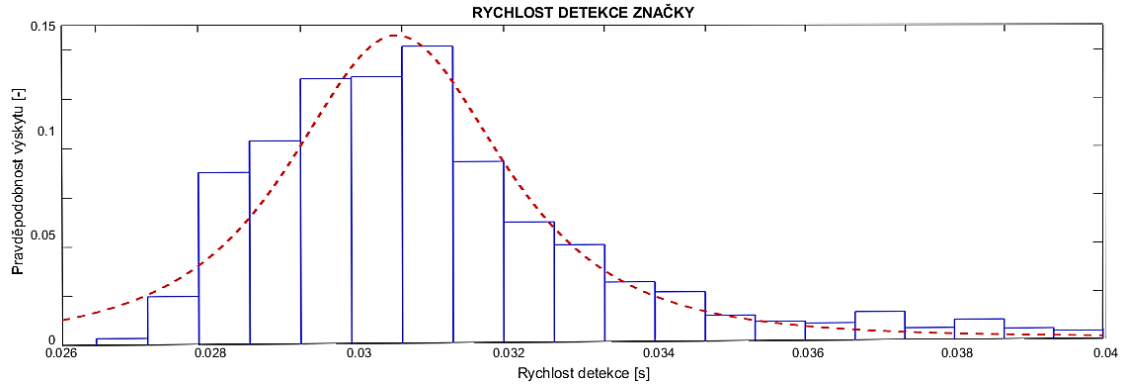
Obr. 30: Ilustrační měření v ose kamery. Jednotlivá měření jsou soustředěná kolem bodů s největší hustotou výskytu (preciznost). Data pro histogram byla zredukována na výběr pětiset prvků.



Obr. 31: Ilustrační měření úhlů na náhodných měřených vzdálenostech. Výběr dat pro histogram opět čítal pět set vzorků.

7.7 Rychlost detekce a spolehlivost měření

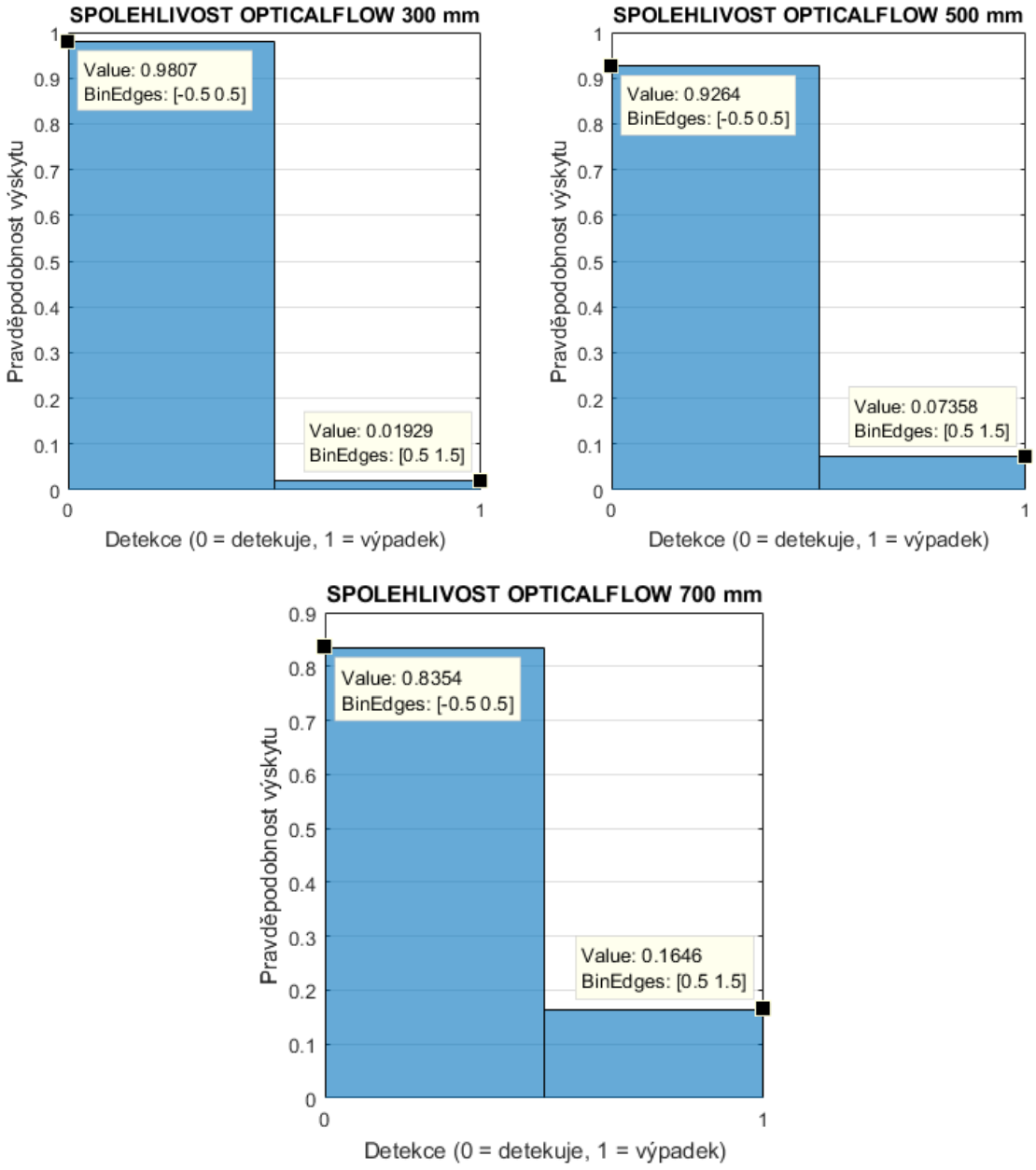
Poslední prováděné měření se zaměřilo na ověření spolehlivosti a rychlosti znovu detekování značky v obraze. Pro účely ověření rychlosti detekce byla do programu vnesena proměnná, která po každé detekci značky opakovaně nastavovala hodnotu proměnné detekující výpadek na hodnotu logické jedničky. Tím bylo dosaženo stavu, kdy měřicí program po každé detekci značky předpokládal její okamžitý výpadek a znovu zahájil fázi detekce. Data z měření času před zahájením a po dokončení detekce byla zaznamenána a zpracována do následujícího grafu (obr. 32).



Obr. 32: Graf zobrazující data z měření rychlosti detekce značky.

Poslední již zmíněné měření se soustředilo na zjištění celkové spolehlivosti sledování značky. Pro tyto účely byl vytvořen jednoduchý test, při kterém se simuloval provoz v semiautonomním konvoji. Test byl rozdělen na tři měřicí vzdálenosti (300, 500 a 700 mm). V těchto určených vzdálenostech byla značka před kamerou přesouvána v horizontálním směru v rozmezí -20 až 20°. Takto byla sledovaná značka před kamerou

přesunuta desetkrát pro každou měřenou vzdálenost. Následující série grafů zobrazuje zpracované výsledky (obr. 33).

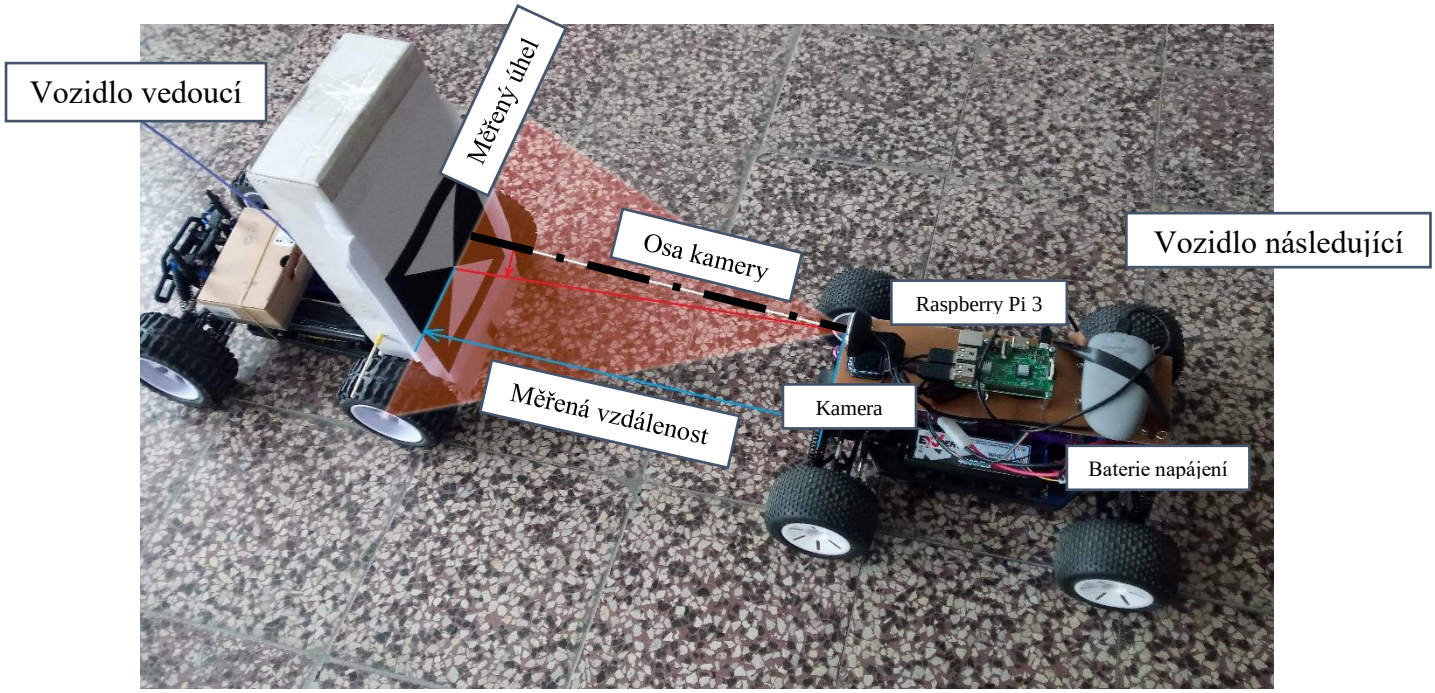


Obr. 33: Grafy se spolehlivostmi metody optického toku Lucas–Kanade pro tři měřené vzdálenosti.

Z naměřených dat tedy vyplývá přibližně devadesátiprocentní spolehlivost měření v uvedeném rozsahu vzdáleností. Tato spolehlivost však platí pouze pro ideální podmínky měření. V podmínkách se zhoršeným osvětlením nebo v prostorách s nainstalovanými zářivkami, které se projevují blikajícím obrazem, nejsou tyto hodnoty spolehlivosti dosažitelné. V těchto prostorách se měření neprovádělo, protože se zde nepředpokládá provoz konvoje.

7.8 Popis výsledného řešení

Řešení bylo realizováno v laboratorních podmínkách na modelu semiautonomního konvoje. Na následující fotografii je shrnut popis samotného modelu a zobrazení měřených rozměrových parametrů (obr. 34).



Obr. 34: Realizace modelu semiautonomního konvoje s parametry.

8 ZÁVĚR

Cílem této práce bylo navrhnout a implementovat systém detekce a sledování pasivního prvku v semiautonomním konvoji. Takto navržené programové řešení dále otestovat v sérii předem zadaných testů, které ověří jeho použitelnost v praktickém nasazení.

Tato bakalářská práce se v druhé kapitola zaměřila na studii aktuálně dostupných technických řešení. Z této rešeršní studie je patrný velmi pokročilý vývoj některých řešení, převážně v oblasti nákladní a vojenské dopravy.

Ve třetí kapitole jsou obecně shrnuty poznatky o senzorech používaných v semiautonomních konvojích. Výběr jednotlivých zařízení vychází z kapitoly předchozí, která pro jednotlivá konkrétní řešení zmiňovala i použité měřicí přístroje.

Ve čtvrté kapitole je krátce shrnut pojem zpracování obrazových dat. Dále je proveden výčet jednotlivých programových knihoven vhodných pro detekci a sledování pasivního prvku na vedoucím vozidle. V samém závěru kapitoly je objasněn důvod pro použití knihovny OpenCV. Tímto důvodem je především rychlost výsledného zkompilovaného programu.

Pátá kapitola je zaměřena na návrh samotného řešení systému. Je zde uveden postup úpravy poskytnuté značky, způsob úpravy obrazových dat za účelem získání aproximovaných kontur obrazce umístěného na detekované značce. Druhá část této kapitoly se zabývá návrhem samotného řešení měření vzdálenosti. Zde bylo použito jednoduché středové projekce, která umožňuje jednoduchým způsobem vypočítat výslednou vzdálenost od kamery. Podobným způsobem je provedeno i měření úhlu mezi středem sledované značky a osou kamery.

Předposlední, šestá část se zaměřuje na praktickou realizaci tohoto řešení. V první části se zmiňuje problematika instalace a integrace knihovny pro zpracování obrazu do prostředí Eclipse. Dále následuje výčet jednotlivých metod zpracování obrazu s popisy jejich funkce. Kapitola je také doplněna o řešení předávání dat dalším částem řízení a o jednoduchou optimalizaci kódu. Optimalizace běhu programu se zaměřila jak na samotný zápis kódu, tak na optimalizaci pomocí kompilátoru GCC.

Poslední kapitola je vyčleněna k popisu provedených měření. Z těchto provedených měření je patrná nutnost použití rychlejšího záznamového zařízení. Dodaná paměťová karta nedosahuje požadovaných stabilních výsledků, a proto se přikročilo k použití techniky ukládání dat do paměti RAM, tzv. ramdisk. Po takto upravené metodě ukládání se podařilo docílit zadaného parametru, a to běhu při stabilních 25 FPS. Tato hodnota je však dosažitelná pouze při ideálních světelných podmínkách. Vedlejším testem při měření hodnoty počtu snímků bylo měření teploty jádra procesoru, které rovněž negativně ovlivňuje tuto veličinu. Měření přesnosti měřicí metody i s výsledky je součástí předposledního měření. Z těchto výsledků vyplývá poměrně dobrá preciznost samotné metody, tato informace ovšem neplatí pro přesnost měřicí metody v měřených vzdálenostech nad 700 mm. Poslední sérií testů byla zjišťována rychlost a spolehlivost detekčního a sledovacího programového řešení. Z těchto měření byla zjištěna velmi rychlá odezva systému na výpadek sledované značky v řádech desítek milisekund

potřebných pro detekci značky. Test spolehlivosti sledování prokázal předpoklad snížení hodnoty spolehlivosti v závislosti na měřené vzdálenosti. Avšak existuje předpoklad použití konvoje s velmi malými rozestupy, pro které je tato spolehlivost pohybující se okolo devadesáti procent dostačující.

Zadané cíle práce byly splněny a navržené řešení je připravené k provozu v jednoduchém semiautonomním konvoji. Mělo by však být doplněno o další metody měření určené pro lokalizaci

9 SEZNAM POUŽITÝCH ZDROJŮ

[1] What is vehicle platooning? *Drivingtests.co.nz* [online]. Auckland (Nový Zéland): Driving Tests Resources, c2017 [cit. 2017-03-20]. Dostupné z: https://www.drivingtests.co.nz/resources/what-is-vehicle-platooning/

[2] *SARTRE* [online]. Cambridge (Velká Británie): SARTRE-Consortium, c2012 [cit. 2017-03-20]. Dostupné z: http://www.sartre-project.eu/en/Sidor/default.aspx

[3] Obr.1 HMI panel. In: *Newatlas.com* [online]. Collingwood (Austrálie): GIZMAG PTY, 2012 [cit. 2017-03-20]. Dostupné z: http://img-1.newatlas.com/volvo-autonomous-7.jpg?auto=format%2Ccompress&fit=max&h=670&q=60&w=1000&s=30109fab3ebc889221eb10b172ab6009

[4] Sartre Vehicle Platooning: First demonstration of SARTRE vehicle platooning. In: *Youtube.com* [online]. Gothenburg (Švédsko): Volvo Car Group, 2011 [cit. 2017-03-20]. Dostupné z: https://www.youtube.com/watch?v=45IRE8W_3L8

[5] The auto pilot for trucks: Highway Pilot. *Daimler.com* [online]. Stuttgart: Daimler, c2017 [cit. 2017-03-20]. Dostupné z: https://www.daimler.com/innovation/autonomous-driving/special/technology-trucks.html

[6] Daimler Tests Self-Driving Truck Platoon in Live Traffic. *Trucks.com* [online]. Florida: TRUCKS.COM INTERNATIONAL, 2016 [cit. 2017-03-20]. Dostupné z: https://www.trucks.com/2016/03/21/daimler-tests-self-driving-truck-platoon-in-live-traffic/

[7] Obr. 2 Reakce na změnu jízdního pruhu. In: *Roadstars.mercedes-benz.com* [online]. Stuttgart: Daimler, c2017 [cit. 2017-03-20]. Dostupné z: https://roadstars.mercedes-benz.com/content/dam/mercedes-benz-trucks/common/events/2015/october/mercedes-benz-actros-with-highway-pilot-world-premiere-on-public-roads/images/content/940/mercedes-benz-actros-with-highway-pilot-world-premiere-on-public-roads-940-04.jpg

[8] European Truck Platooning. *European Truck Platooning* [online]. Amsterdam: Conference of European Directors of Roads, c2016 [cit. 2017-03-20]. Dostupné z: https://www.eutruckplatooning.com

[9] EcoTwin se zúčastní testu European Truck Platooning Challenge. *Napatrucks.cz* [online]. Pardubice: NAPA TRUCKS, c2017 [cit. 2017-03-20]. Dostupné z: http://napatrucks.cz/novinka-ecotwin-se-zucastni-testu-european-truck-platooning-challenge

[10] Obr. 3 Vozidla DAF EcoTwin. In: *Eutruckplatooning.com* [online]. Amsterdam: European Truck Platooning, c2016 [cit. 2017-03-20]. Dostupné z: https://www.eutruckplatooning.com/Press/Photos+DAF/93173.jpg?Width=1200&Height=-1

[11] MAN Truck2Truck Platooning Challenge. *Ukhaulier.co.uk* [online]. Felixstowe (Velká Británie): UK Haulier, 2016 [cit. 2017-03-20]. Dostupné z: https://www.ukhaulier.co.uk/news/road-transport/platooning/man-truck2truck-platooning-challenge-europe/

[12] The Platooning Experience. *Peloton-tech.com* [online]. California: Peloton Technology, c2017 [cit. 2017-03-20]. Dostupné z: http://peloton-tech.com/how-it-works/#1452644330761-1e5676cf-7227

[13] Peloton Technology. *Crunchbase.com* [online]. San Francisco: Crunchbase, c2017 [cit. 2017-03-20]. Dostupné z: https://www.crunchbase.com/organization/peloton-technology#/entity

[14] Autonomous Mobility Applique System (AMAS): Overview. *Lockheedmartin.co.uk* [online]. Bethesda (Spojené státy americké): Lockheed Martin Corporation, c2017 [cit. 2017-03-20]. Dostupné z: http://www.lockheedmartin.co.uk/us/products/amas1/mfc-amas-overview.html

[15] Obr. 5 Vozidlo M915. In: *Defense-update.com* [online]. Qadima (Izrael): Defense-Update, c2002-2017 [cit. 2017-03-20]. Dostupné z: http://defense-update.com/wp-content/uploads/2014/02/m915_amas.jpg

[16] Obr. 6: Ukázka použití radaru. In: *Swedespeed.com* [online]. Toronto: Vortex Media Group, 2010 [cit. 2017-03-23]. Dostupné z: http://www.swedespeed.com/artman2/uploads/1/pedestrian-detection-animation.jpg

[17] Differential Global Positioning System. *Australian Government: Australian Maritime Safety Authority* [online]. Canberra (Austrálie): Australian Maritime Safety Authority, c2017 [cit. 2017-03-23]. Dostupné z: https://www.amsa.gov.au/navigation/services/dgps/

[18] Obr. 8: Obrázek ilustruje výstupní data. In: *Uidaho.edu* [online]. Moscow, Idaho (Spojené státy americké): University of Idaho, c2017 [cit. 2017-03-23]. Dostupné z: http://www.webpages.uidaho.edu/vakanski/Images/Lidar_Map_2.jpg

[19] OpenCV library. *Opencv.org* [online]. San Francisco: OpenCV team, c2017 [cit. 2017-03-23]. Dostupné z: http://opencv.org

[20] *EmguCV: OpencV in .NET* [online]. Markham (Kanada): Emgu Corporation, c2017 [cit. 2017-03-23]. Dostupné z: http://www.emgu.com/wiki/index.php/Main_Page

[21] Intel® Integrated Performance Primitives (Intel® IPP). *Software.intel.com* [online]. Santa Clara (Spojené státy americké): Intel Corporation, c2017 [cit. 2017-03-23]. Dostupné z: https://software.intel.com/en-us/intel-ipp

[22] *SimpleCV* [online]. San Francisco: Sight Machine, c2013 [cit. 2017-03-23]. Dostupné z: http://simplecv.org

[23] *BoofCV* [online]. Pittsburgh: Abeles, Peter, c2017 [cit. 2017-03-23]. Dostupné z: http://boofcv.org

[24] *MathWorks* [online]. Boston: The MathWorks, c1994-2017 [cit. 2017-03-23]. Dostupné z: https://www.mathworks.com

[25] Obr. 9: Graf z testů výkonosti. In: *Packtpub.com* [online]. Birmingham (Spojené království): Packt Publishing Limited, c2017 [cit. 2017-03-23]. Dostupné z: https://www.packtpub.com/graphics/9781783559527/graphics/9527OT_1_1.jpg

[26] Line Detection. *Homepages.inf.ed.ac.uk* [online]. Edinburgh (Spojené království): Hypermedia Image Processing Reference, c2003 [cit. 2017-03-23]. Dostupné z: http://homepages.inf.ed.ac.uk/rbf/HIPR2/linedet.htm

[27] HARTLEY, Richard. a Andrew. ZISSERMAN. *Multiple view geometry in computer vision*. 2nd ed. New York: Cambridge University Press, 2003. ISBN 978-052-1540-513.

[28] Vybíráme vhodné čočky pro webkamery. *Netcam.cz* [online]. Chrudim: BLUECOM, c2017 [cit. 2017-03-23]. Dostupné z: http://www.netcam.cz/encyklopedie-ip-zabezpeceni/vybirame-vhodne-cocky.php

[29] *Raspberrypi.org* [online]. Cambridge (Spojené království): RASPBERRY PI FOUNDATION, c2017 [cit. 2017-03-23]. Dostupné z: https://www.raspberrypi.org

[30] Opencv Documentation. *Docs.opencv.org* [online]. San Francisco: OpenCV team, c2011-2014 [cit. 2017-03-23]. Dostupné z: http://docs.opencv.org/2.4.13.2/index.html

[31] *Eclipse.org* [online]. Ottawa: The Eclipse Foundation, c2017 [cit. 2017-03-23]. Dostupné z: https://eclipse.org

[32] BRADSKI, Gary R. *Learning OpenCV*. Sebastopol (Spojené státy americké): O'Reilly, c2008. ISBN 978-059-6516-130.

[33] FEINEN, Christian. *Object Representation and Matching Based on Skeletons and Curves*. Berlin: Logos Verlag Berlin, c2016. ISBN 978-383-2542-573.

[34] *Sledování trajektorie*. Brno, 2009. Bakalářská práce. Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Miloslav Richter.

[35] ROJAS, Raúl. *Lucas-Kanade in a Nutshell*. Berlin, c2017. Dostupné také z: http://www.inf.fu-berlin.de/inst/ag-ki/rojas_home/documents/tutorials/Lucas-Kanade2.pdf

[36] Introducing JSON. *Json.org* [online]. Geneva: Ecma International, c2013 [cit. 2017-03-23]. Dostupné z: http://www.json.org/index.html

[37] *Tips for Optimizing C/C++ Code*. Clemson (Spojené státy americké), 2014. Dostupné také z: https://people.cs.clemson.edu/~dhouse/courses/405/papers/optimize.pdf

[38] *GCC, the GNU Compiler Collection* [online]. Boston: Free Software Foundation, c2017 [cit. 2017-03-23]. Dostupné z: https://gcc.gnu.org

[39] *LifeCam HD-3000: Technical Data Sheet*. Redmond , c2017. Dostupné také z: http://download.microsoft.com/download/0/9/5/0952776D-7A26-40E1-80C4-76D73FC729DF/TDS_LifeCamHD-3000.pdf

[40] How to Create a RAM Disk in Ubuntu Linux: (15.04, 15.10…). *Hecticgeek.com* [online]. HecticGeek, 2015 [cit. 2017-03-23]. Dostupné z: http://www.hecticgeek.com/2015/12/create-ram-disk-ubuntu-linux/

[41] EMAMI, Shervin a Kaehler ADRIAN. *Mastering OpenCV with Practical Computer Vision Projects*. Birmingham (Spojené království): Packt Publishing, 2012. ISBN 9781849517836.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

A. OBSAH PŘILOŽENÉHO CD

Název	**Popis**
BS_Kaura.pdf	Textová část bakalářské práce.
Testovani.mp4	Videozáznam z testování systému bez trasovacího algoritmu.
Optical Flow	Zdrojové kódy k detekčnímu algoritmu.
UkazkaBehu.mp4	Videozáznam z běhu detekčního programu.